UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Politècnica Superior d'Enginyeria
de Manresa

Treball Final de Grau

METEOR

Eina per el desenvolupament d'aplicacions web en temps real

Grau en Enginyeria de Sistemes TIC

Curs 15/16

Autor: Oriol Vall Hernández
Directora: Rosa Giralt Mas
Data: 01/07/2016
Localitat: Manresa

Dedicatoria

RESUM

Els frameworks web permeten crear webs complexes amb relativa facilitat. En aquest treball es fa una recerca sobre el funcionament del framework “Meteor”, un framework full-stack en JavaScript. Dins la recerca, s’estudia quins elements composen aquest framework en el seu interior, quin tipus de comunicació utilitza entre el client i el servidor i s’expliquen algunes eines externes per tal de millorar les aplicacions creades amb aquest framework.
Després de la recerca, s’han creat varies aplicacions d’exemple per demostrar el potencial d’aquesta eina, com per exemple: una aplicació per gestionar les despeses o una plataforma per fer el seguiment de projectes d’enginyeria.

ABSTRACT

Frameworks web allow to create web pages with huge complexity really easy. This project make a research about how Meteor works, a full-stack JavaScript framework. The research study which elements compose this framework, which kind of protocol is used to communicate the server and the client and explain some external tools to be able to improve apps created with this framework.
After the research, some apps were created to show the possibilities of this tool. for example, we made an app to manage group expenses or another app to track engineering projects.

ÍNDEX

1. INTRODUCCIÓ

Des de la creació del “World Wide Web” el 1989 han existit les pàgines web. Al principi eren simples pàgines estàtiques escrites en HTML. A mesura que milloraven les infraestructures i apareixien noves necessitats per els usuaris, es van anar creant noves versions de HTML afegint noves funcionalitats junt amb nous llenguatges com JavaScript el 1995 (sota el nom de “Mocha”). Amb totes les noves funcionalitats va aparèixer la necessitat de simplificar el procés de creació de pàgines web i, d’aquesta manera, van aparèixer els frameworks.

Actualment, els frameworks per la creació de webs poden incloure eines per treballar tant en el servidor com en els navegadors dels clients. A més, amb la creació de les xarxes socials, ha aparegut la necessitat de crear aplicacions web amb la funcionalitat de ser “real time”. Els usuaris volen veure que les seves accions tenen una repercussió immediata a l’aplicació.

Per la importància dels frameworks en l’actualitat, en aquest projecte es fa un primer estudi sobre els frameworks actuals i es fa una petita comparació entre ells. Aleshores ens centrarem en un d’ells, un nou framework full-stack (servidor i client) en JavaScript anomenat Meteor. Durant l’estudi del framework anomenat Meteor s’expliquen les seves caracteristiques, el seu funcionament intern (Base de dades, comunicació client-servidor, ...) i s’anomenen algunes eines per complementar-lo. La segona part del projecte consta de la creació de varies aplicacions utilitzant aquesta tecnologia per demostrar el seu potencial.

En aquest projecte es comentarà codi en HTML i JavaScript, per aquest motiu, es recomana tenir unes nocions bàsiques d’aquest llenguatge per tal de poder entendre’l completament.

2. FRAMEWORKS WEB

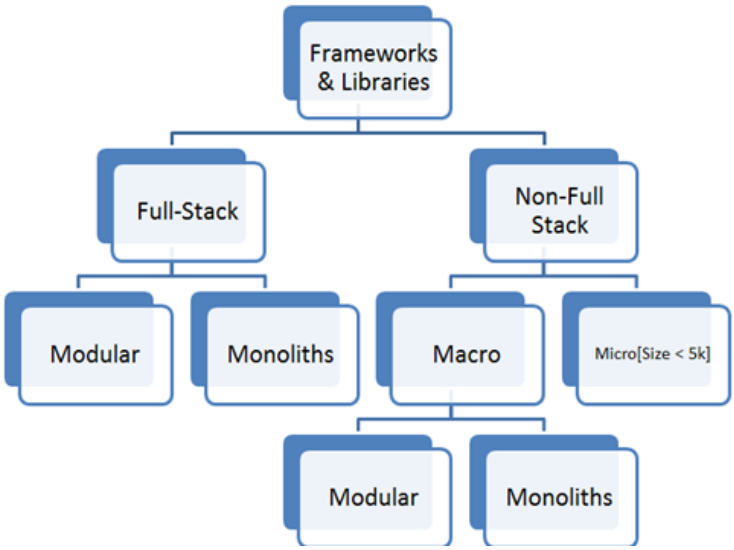
Un framework és una llibreria de codi que facilita la feina als desenvolupadors quan dissenyen aplicacions web.

Els frameworks webs engloben el coneixement dels últims vint anys en el desenvolupament de pàgines i aplicacions web. Al mateix temps, faciliten la reutilització de codi per les operacions HTTP i l'estructuració del codi per tal que futurs desenvolupadors amb coneixement del framework puguin millorar i mantenir l'aplicació.

Algunes de les funcionalitats que poden incloure els frameworks són:
- Manipulació de bases de dades
- Adreçament de URLs
- plantilles amb format HTML, XML o JSON entre altres

Els frameworks es poden classificar en dos grans grups:
- Full-Stack: Contenen eines per treballar tant en el client com en el servidor.
- Non-Full-Stack: Tots aquells que no són Full-Stack. Dins d'aquest grup es divideixen en funció de la seva mida, poden ser Micro (<5k) o Macro Frameworks.

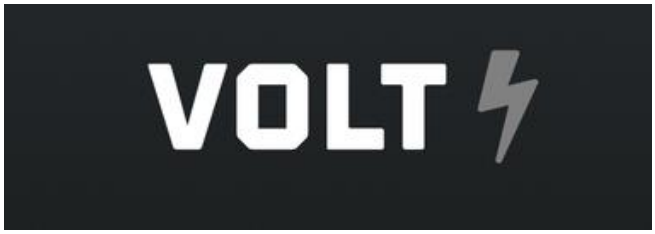
Tot allò que no sigui Micro Framework es pot dividir en dues subclasses:
- Modulars
- Monolítics



Imatge 1: Esquema de clasificació dels frameworks

A continuació anem a comentar alguns dels frameworks més coneguts o interessants. Els classificarem en funció del llenguatge en que estàn escrits (Ruby, JavaScript o Python) i descriurem els advantatges o inconvenients que tenen respecte la resta.

2.1. RUBY FRAMEWORKS



Imatge 2: Logotip de Volt

Volt és un framework similar a Meteor en molts aspectes, bàsicament perquè els dos intenten solucionar problemes similars. Els dos frameworks utilitzen el mateix llenguatge en el servidor i en el client, els dos mantenen una connexió permanent per actualitzar les aplicacions web en temps real i els dos utilitzen llenguatge de templates en els fitxers HTML entre altres semblances.

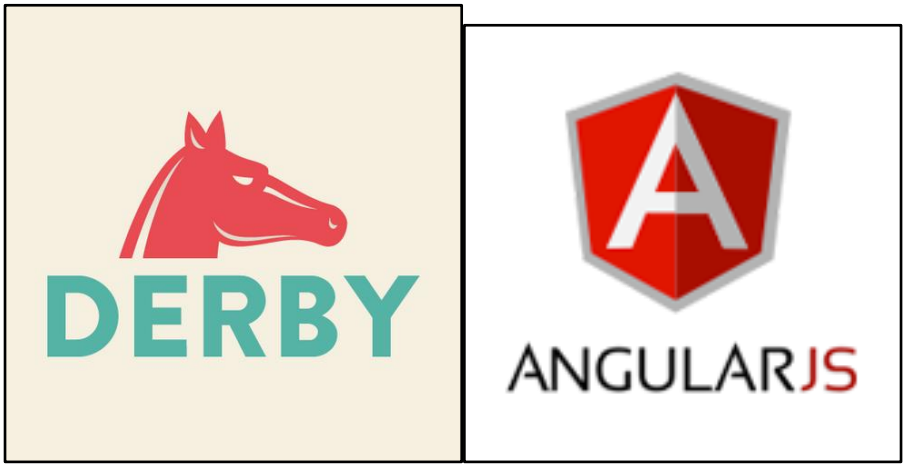
Imatge 3: Logotip de Ruby on Rails

Ruby on Rails és un altre framework bastant conegut. La seva versió 1.0 va ser publicada el 2005. Aquest framework utilitza el model MVC (Model-View-Controller) per tal d'organitzar les seves aplicacións.
Aquest framework va incorporar la filosofia de DRY (Don't repeat yourserf) que significa que les definicións s'han de fer un sol cop, aleshores el mateix framework s'encarrega d'enllaçar-

ho tot a partir de la seva base de dades interna. Aquest mètode de DRY també ha estat incorporat en el framework de Meteor.

La diferencia més important d'aquests frameworks amb Meteor és el llenguatge de programació que utilitzen, **Ruby** en comptes de JavaScript. Això fa la programació més fàcil si es tenen conceptes previs amb Ruby.
Un altre diferencia important entre ells és la **gestió de dades**. Per exemple, Meteor utilitza el mètode de publicacions i subscripcions mentre que en Volt les dades es carreguen reactivament per tot allò que està sent observat.

2.2. JAVASCRIPT FRAMEWORKS



Imatge 4: Logotips de DerbyJS i AngularJS

Imatge 5: Logotips de SailsJS i Meteor

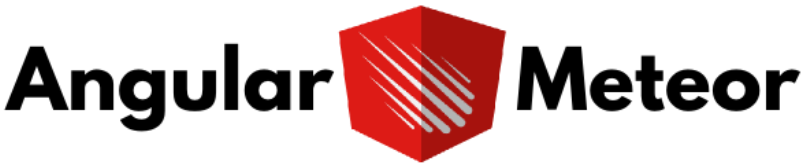
Aquests frameworks estan tots escrits en JavaScript, només per això, observant el codi d'una app d'aquests frameworks ens poden semblar totes molt similar.

Entre els possibles frameworks en JavaScript, el més famós seria AngularJS (desenvolupat per Google). Tot i ser un framework molt utilitzat, AngularJS és només client-side framework. Per això, es sol utilitzar juntament amb MongoDB i Node.js (amb el server-side framework: Express.js) creant l'anomenat **MEAN stack.**

Imatge 6: Logotip de MEAN stack (MongoDB + Express.js + AngularJS + Node.js)

El principal avantatge al utilitzar Meteor és que tens totes aquestes funcionalitats automaticament i sense configuració extra.

Un altre punt important és la possible utilització de AngularJS junt amb Meteor (anomenat **Angular-Meteor**).



Imatge 7: Logotip Angular-Meteor

El principal avantatge que tenen els frameworks en JavaScript per sobre dels altres és que fan possible la programació tant del client com del servidor amb un sol llenguatge.

El que fa destacar Meteor per sobre d'altres frameworks en JavaScript és el fet que permet crear prototips de forma molt ràpida. També té la característica de ser multiplataforma, pots dissenyar una aplicació perquè funcioni per navegadors i també de forma nativa en dispositius mòbils.

En comparació amb el següent framework full-stack en JavaScript (DerbyJS), Meteor supera de molt els articles i projectes a StackOverflow i GitHub, això mostra que aquest framework té una gran comunitat i bastant activa al darrera.

Metric	Meteor	Derby
StackOverflow questions	21298	101
GitHub stars	33939	4031
GitHub commits	15889	1688
GitHub forks	4168	228
Twitter followers	54863	2107
Shares on LinkedIn	625	29

**Dades del Maig de 2016*

Tot i la gran diferència a comunitat d'usuaris, els altres frameworks proporcionen algunes millores respecte Meteor:

- carregar les pagines en el servidor i no en el client.
- optimitzar les pàgines per motors a de cerca (SEO).
- Poden utilitzar altres bases de dades per defecte (MySQL per exemple).

Un altre problema que trobem, és que els altres frameworks encara es troben en versió Beta, DerbyJS està en la versió 0.8.x, mentre que SailsJS encara està per la v0.12.

2.3. PYTHON FRAMEWORKS

Imatge 8: Logotips Django i Web2Py

En el cas que es busqui algun framework en llenguatge Python hi ha dues opcions importants: Django i Web2Py.

En comparació amb Meteor, on la seva primera release va ser llençada el 2012, Django i web2py van començar el 2005 i el 2007 respectivament.
Aquest fet es veu reflectit en la maduresa del frmawork, ja que hi ha funcionalitats que Meteor encara no ha implementat (com la incorporació de BD relacionals). Peró tot i així hi ha aspectes en que Meteor segueix tenint avantatges respecte la competència.

Per exemple, la funcionalitat *real-time* característica de Meteor és gràcies al seu nucli en Node.js, MongoDB i el protocol DDP, tres característiques que no ofereixen els frameworks en python. La facilitat de programar en un sol llenguatge tampoc la podem aconseguir amb els frameworks en python, ja que per programar el *frontend* és necessari la presencia de JavaScript (*jQuery* per exemple).

3. METEOR

En aquest projecte s'ha decidit investigar i treballar sobre Meteor, un dels frameworks en JavaScript, degut al seu increment de popularitat Durant els últims mesos i la seva caracteristica de treballar en temps real que el fa tant especial.

3.1. QUE ÉS METEOR?

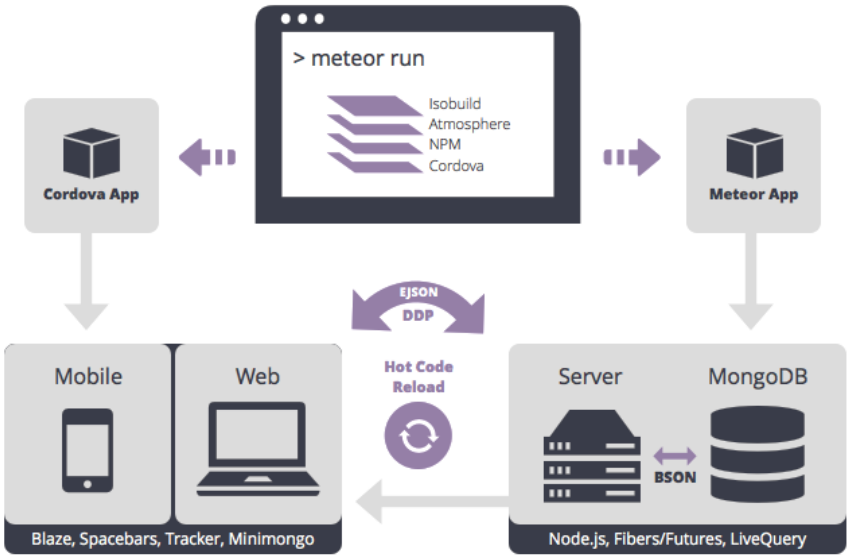
Meteor es un framework de codi obert escrit utilitzant Node.js (JavaScript) desenvolupada per "Meteor Development Group".

Imatge 9: Logotip Meteor

Meteor va formar part del programa "Y Combinator" (empresa acceleradora de startups tecnològiques) el 2011, on va aconseguir una inversió de 20M de dòlars.

Bàsicament, Meteor es un framework "full-stack", això vol dir que podem treballar en front-end i back-end tant en el client com en el servidor i això permet fer web dinàmiques amb temps real de forma ràpida i senzilla, ja que ho tenim tot centralitzat en el mateix lloc. Per fer-ho, meteor ha fusionat llibreries i extensions per fer la programació més senzilla. Podríem dir que les parts més importants de Meteor són les següents:

- Node.js: Servidor en JavaScript amb alta escalabilitat.
- MongoDB: Base de dades per defecte, no relacional (NoSQL).
- Minimongo: Copia de la base de dades en el client.
- Apache Cordova: Framework capaç de produir codi natiu per a SO mòbils (per exemple: Android o iOS).
- SpaceBars: Llenguatge per els "templates" en HTML basat en Handlebars / Moustache (Logic-less temaplates).
- Sistema de subscripcions per a les actualitzacions dels clients.
- Comunicació client - servidor amb el protocol DDP (Distributed Data Protocol).



Imatge 10: Diagrama d'una aplicació programada amb Meteor

3.1.1. Punts forts

Meteor s'ha fet molt popular pel fet de que les seves aplicacions són en **temps real** per defecte, això és una característica important ja que d'aquí poc els usuaris esperaran aplicacions web per treballar instantàniament.

Un altre atractiu de Meteor és la seva programació en **un sol llenguatge** (JavaScript) tan en el client com en el servidor. Això permet més rapidesa alhora de desenvolupar l'aplicació i redueix els errors a causa d'utilitzar varis llenguatges en un mateix projecte.

Un altre manera que utilitza Meteor per tal d'estalviar temps al programador és la utilització de **paquets** que inclouen funcionalitats extra simplement amb una comanda per tal d'incloure el paquet en el projecte. Els paquets es poden buscar des de la pàgina web d'Atmosphere (atmospherejs.com).

Meteor no obliga a utilitzar cap estructura de fitxer (com podria ser MVC), el que si que trobem són unes normes que segueix Meteor en el moment de carregar una aplicació, algunes d'elles són les següents:

- La carpeta **/lib** sempre es carrega en primer lloc.
- La carpeta **/client** només serà executat en el client.
- La carpeta **/server** només serà executat en el servidor.
- Els arxius dins de la carpeta **/private** només podran ser accessibles per codi executat en el servidor.
- Els arxius dins la carpeta **/public** seran enviats als visitants, aquí hi hauran arxius com imatges o favicons.
- Els arxius **main.*** sempre seran els últims en carregar-se.
- La resta d'arxius i carpetes es carregaran de forma alfabètica.

Per tant, seguint una mica les normes, he decidit estructurar els meus projectes de la següent manera:

```
Projecte
├── client
│   ├── controllers
│   │   └── Controllers.js
│   └── views
│       └── Files.html
├── lib
│   └── Router.js
├── server
│   └── Data.js
└── models
```

I les funcions dels fitxers serien les següents:

- **Router.js** : És l'encarregat de gestionar les URLs i decidir quin fitxer html servir a cada client utilitzant el paquet **Iron.Router**.
- **Data.js** : Aquí hi hauria les dades que haurien d'estar al servidor, les guardaríem utilitzant la base de dades **MongoDB**.
- **Files.html** : A la banda del client hi guardaríem tots els fitxers HTML per crear les pàgines web.
- **models/** : en aquesta carpeta hi haurà arxius en JavaScript on crearem els objectes necessaris per el nostre projecte amb les seves respectives funcions.
- **Controllers.js** : Aquests fitxers serien els encarregats de gestionar totes les funcions dels fitxers HTML utilitzant el llenguatge JavaScript.

Per demostrar el gran potencial que té aquesta plataforma podem mirar algunes aplicacions web creades amb Meteor:

- Telescope (www.telescopeapp.org): App per crear comunitats pròpies (com per exemple crater.io, comunitat per discutir sobre Meteor).
- Rocket.Chat (rocket.chat): Chat web de codi obert.
- ThingStudio (www.thingstud.io): App per crear interfícies orientades a aplicacions sobre IoT (Internet of Things).
- Wekan (wekan.io): App de codi obert per administrar feines (similar a "trello").

3.1.2. Instal·lació i ús

Per tal d'instal·lar Meteor en un SO Linux o iOS, heu d'executar la següent comanda en una terminal:

```
curl https://install.meteor.com/ | sh
```

En el cas de SO Windows, s'ha de descarregar l'instal·lador des de la pàgina oficial:

```
https://www.meteor.com/install
```

Un cop instal·lat, simplement hem d'obrir una terminal i escriure la següent comanda per crear un nou projecte:

```
meteor create NewProject
```

Automaticament es crearà un directori anomenat '**NewProject**' amb tots els fitxers que necessita l'aplicació Meteor i un petit projecte funcional.

```
NewProject/
├── NewProject.css
├── NewProject.html
├── NewProject.js
└── .meteor
    └── ...
```

Per executar el servidor simplement ens hem d'ubicar dins la carpeta del projecte i executar la comanda de meteor:

```
cd NewProject
meteor run
```

L'últim pas seria visitar la URL on el servidor està escoltant, per defecte '**http://localhost:3000**'.

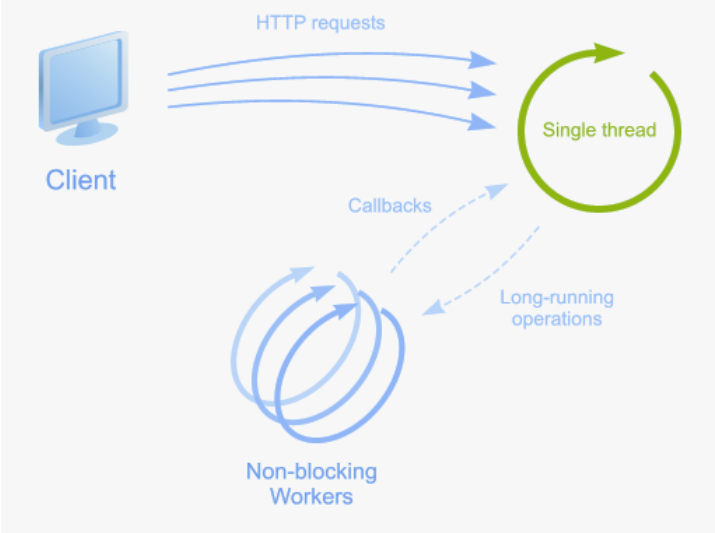
3.2. SERVIDOR

3.2.1. Node.JS

Podríem dir que Node.js és un entorn JavaScript de baix nivell que proporciona les funcions per rebre i enviar peticions en HTTP, per tant, podríem dir que Node és el cor de Meteor.

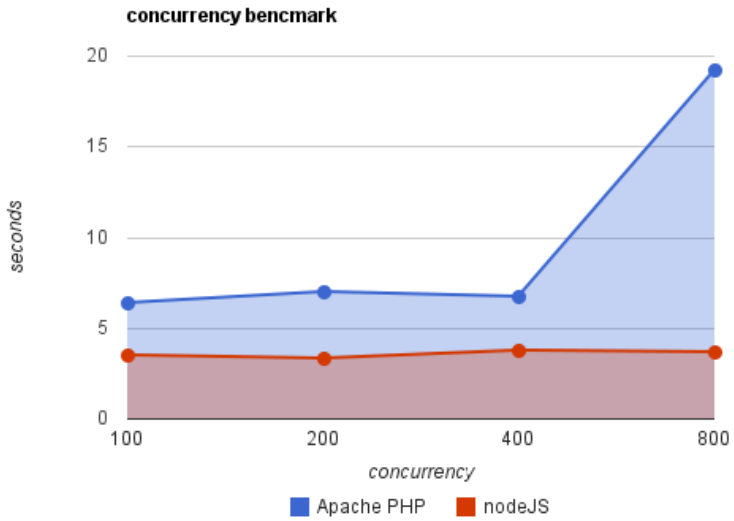
Imatge 11: Logotip Node.js

Node.js consisteix en el motor de JavaScript V8 de Google i està dissenyat per tal de programar el client i el servidor en JavaScript, això implica més facilitat alhora de programar, ja que només hem d'aprendre un sol llenguatge. Node utilitza un model asíncron i dirigit per esdeveniments. A més, Node ha estat pensat per tal de poder respondre a moltes connexió simultànies, per fer-ho, es programa en un sol fil amb un temps de resposta molt petit i en el cas que hi hagi alguna aplicació bloquejat, I/O per exemple, Node creara un segon fil en segon pla.



Imatge 12: Diagrama de funcionament d'un servidor Node.js

En comparació amb altres servidors web, com per exemple Apache, es pot percebre la millora, ja que aquest últim crea un fil per cada nova connexió.



Imatge 13: Comparació de rendiment entre Apache i Node.js

Un altre punt fort de Node.js és la simplicitat alhora de programar, per exemple, utilitzant Node.js podem crear un servidor funcional amb nomes 5 línies de codi.

```
my_http = require("http");
my_http.createServer(function(request,response){
    response.writeHeader(200, {"Content-Type": "text/html"});
    response.end("Oriol Vall\n");
}).listen(8080,"127.0.0.1");
```

Algunes companyies importants que utilitzen Node.js són Netflix, PayPal o LinkedIn.

3.2.2. MongoDB

Per tal de poder guardar dades en el servidor, Meteor incorpora MongoDB com a base de dades per defecte.

Imatge 14: Logotip MongoDB

MongoDB és una base de dades noSQL, per tant no tenim taules ni registres. Aquest tipus de base de dades orientada a documents i organitzada en col·leccions. Els documents són guardats en BSON (representació binaria de JSON).

Exemple dades en format JSON:

```
{
        Nom: "Oriol",
        Edat: 21,
        Amics: [
        {
                Nom: "Oscar",
                Edat: 24
        },
        {
                Nom: "Adrià",
                Edat: 21
        }
        ]
}
```

La principal diferencia respecte a les bases de dades relacionals és que no es necessari seguir cap esquema. Els documents d'una mateixa col·lecció poden tenir diferents esquemes.

MongoDB ha estat implementat en diverses webs i serveis com ara Craiglist, eBay, Foursquare i el New York Times entre d'altres.

Per crear una col·lecció s'utilitza la següent comanda:

```
Name = new Mongo.Collection('Name');
```

Algunes de les funcions disponibles més importants per operar amb la base de dades són:

- insert() : Afegeix un document o varis documents a una col·lecció.
- find() : Selecciona documents d'una col·lecció i retorna el cursor dels elements seleccionats.
- update() : Modifica un document o varis documents existents en una col·lecció.
- remove() : Esborra tots els documents d'una col·lecció que concordin amb un filtre.
- drop() : Esborra una col·lecció.

Les operacions a la base de dades es faran de manera asíncrona, per tant el següent codi és possible sense necessitat de callbacks:

```
Name.insert({ _id :  'myName' });

const var = Name.findOne({ _id : 'myName' });

console.log(var);
```

3.3. CLIENT

3.3.1. Blaze

Blaze és el sistema de presentació del frontend que utilitza Meteor per crear les interfícies d'usuari per defecte. Altres sistemes que poden ser utilitzats són *React* o *Angular*.

Per tal d'aconseguir una bona UI (*User Interface*), Blaze utilitza Spacebars junt amb codi JavaScript per gestionar totes les funcionalitats.

SpaceBars en un llenguatge per les plantilles (*templates*) de Meteor inspirat en Handlebars. Aquest llenguatge facilita la programació en HTML i proporciona noves funcionalitats, com per exemple poder utilitzar condicionals o bucles dins d'un fitxer HTML.

Les expressions de Spacebars en fitxers HTML estan delimitades per "**{{**" i "**}}**".
Les etiquetes personalitzades han de ser creades en un fitxer de la següent manera:

```
Template.llista.helpers({
	persones: [
		{ nom: "Oriol", edat: 21, estat: true},
		{nom: "Oscar", edat: 24, estat: false}
	];
});
```

Algunes eines que inclou Spacebars són:

- **each:** Itera sobre un Array/Llista o un Cursor (return d'un *find* a una col·lecció de MongoDB)
- **with:** S'utilitza per entrar dins d'un objecte del JSON i així fer el codi més net i estructurat. Per tant, en comptes d'escriure '*root.element*' només faria falta escriure '*element*'.
- **if, else:** Operador condicional típic.
- **unless:** Operador oposat de *'if'* (=if not).
- **> :** Eina per adjuntar altres templates. Normalment s'utilitza dins d'un block *'each'.*

Les eines condicionals i bucles s'han d'obrir amb "**#**" i tancar amb "***/***".

El següent codi es un exemple d'un fitxer HTML utilitzant el helper anterior:

```
<template name="element_llista">
	{{#if estat}}
		{{nom}} - {{edat}}
	{{/if}}
</template>

<template name="llista">
	{{#each persones}}
		{{> element_llista}}
	{{/each}}
</template>
```

Resultat d'executar el codi anterior:

```
Oriol - 21
```

Per ajudar als fitxers HTML, Meteor utilitza codi en JavaScript.
En les aplicacions realitzades en aquest projecte el que s'ha utilitzat més són els elements *'helpers'* i *'events'.*

Els *helpers* són els encarregats de proporcionar les dades des de les bases de dades o dades estàtiques per estalviar línies de codi en els fitxers HTML i aconseguir un codi més simple.

El següent codi és un *helper* que retorna una *query* de Mongo amb tots els usuaris:

```
Template.test.helpers({
  users: function(){
    return Meteor.users.find();
  }
});
```

Els elements *events* serien els 'actuadors', aquest codi s'encarrega de decidir que fer quan es detecta algun canvi per part de l'usuari (clicar botons, links o prémer alguna tecla per exemple).

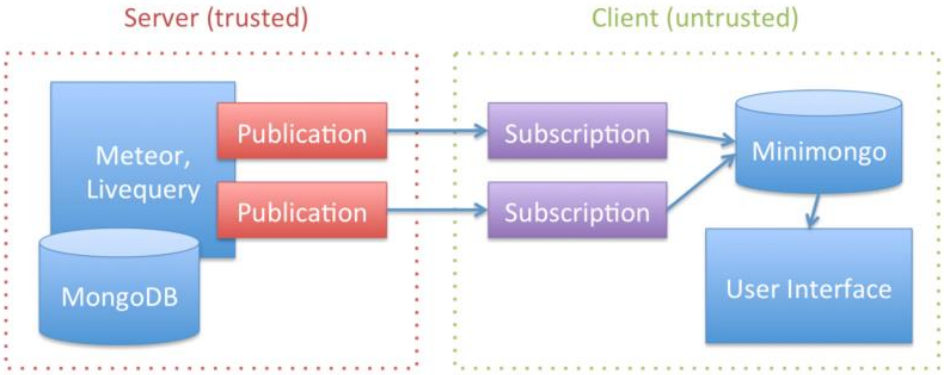
A continuació hi ha un exemple de *event* que s'utilitza per retrocedir a la URL anterior quan es clica el botó amb **id=cancel:**

```
"click #cancel" : function(event, template){
    history.back();
  }
```

3.3.2. MiniMongo

Aquest paquet es una reimplementació de tota (o quasi tota) la MongoDB API. És com un emulador de la base de dades funcionen dins del navegador web del client.

Dins d'aquesta base de dades, hi haurà una copia de les subscripcions (col·leccions de dades que volem tenir actualitzades) dels clients. Això permet poder fer operacions dins d'aquesta base de dades i tenir una resposta instantània, cosa que seria impossible si totes les operacions es sol·licitessin a la base de dades MongoDB del servidor, ja que tindríem un retard bastant important.



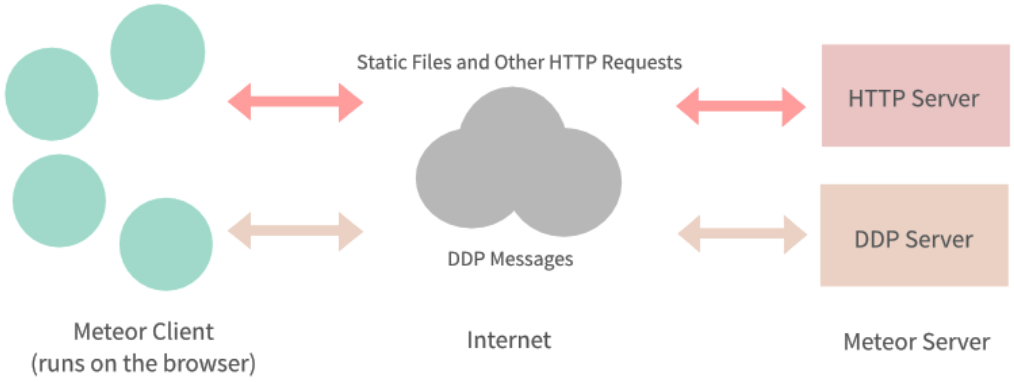
Imatge 15: Diagrama de publicacions - subscripcions d'un servidor a un client Meteor

3.4. COMUNICACIÓ CLIENT - SERVIDOR

La comunicació entre el client i el servidor es duu a terme de forma automàtica, per tant el programador no s'ha de preocupar per aquest aspecte.

Peró per entendre com funciona tot, explicarem com es produeix aquesta comunicació. El primer que hem d'entendre és que Meteor pot rebre 3 sol·licituds diferents: Fitxers estàtics, Missatges DDP i sol·licituds HTTP.

Encara que Meteor treballi en un sol port, dins de Meteor hi trobem dos servidors, un servidor DDP i un servidor HTTP per organitzar totes sol·licituds.



Imatge 16: Esquema de la comunicació client - servidor

3.4.1. Fitxers estàtics i HTTP

Els fitxers estàtics són tots aquells ubicats dins la carpeta "/public" (bàsicament seran totes les imatges).
Meteor serveix els arxius estàtics en el moment en que s'inicia l'aplicació. A més, Meteor concatena tot el codi JavaScript (incloent els templates) i els arxius CSS i els serveix com a fitxers estàtics.

Meteor pot gestionar les sol·licituds en HTTP de la mateixa manera que les aplicacions tradicionals. Per exemple, per pujar un arxiu a una aplicació Meteor s'utilitzaran sol·licituds HTTP.

El servidor encarregat de gestionar els arxius estàtics i les sol·licituds HTTP és el servidor HTTP. Meteor utilitza el mòdul "connect" de Node.js per aquesta tasca.

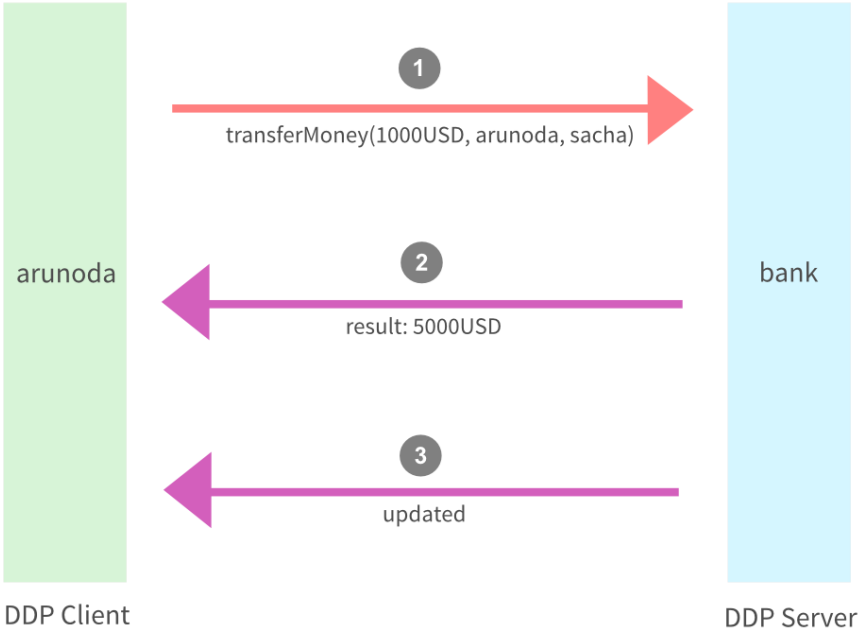
3.4.2. DDP (Distributed Data Protocol)

DDP és protocol simple que utilitza Meteor per comunicar-se entre el client i el servidor. Totes les dades de subscripcions - publicacions (bases de dades), mètodes i operacions amb MongoDB són enviades a traves de missatges DDP.

Tots els missatges DDP són gestionats per el servidor DDP que, de fet, és una modificació d'un servidor SockJS (llibreria que proporciona WebSockets per Node.js) per Meteor.

La implementació de missatges DDP és simple, es tracta de varis missatges JSON (concretament EJSON) dins d'un WebSocket.

El format EJSON és una extensió de JSON que inclou el format de dades 'Date' i 'Binary'. També permet definir formats personalitzats per l'usuari a més inclou la opció *'canonical'* que assegura la mateixa sortida si les dades d'entrada són iguals (cosa que amb JSON no es podia assegurar ja que JSON permet a les claus d'un objecte aparèixer en qualsevol ordre).



Imatge 17: Exemple de comunicació DDP entre client i servidor en una transacció

Els missatges en format DDP de la comunicació anterior tindrien un aspecte com aquest:

```
1. { "msg" : "method", "method" : "transferMoney", "params" : ["1000USD", "arunoda",
"sacha"], "id" : "randomId-1" }

2. { "msg" : "result", "id" : "randomId-1", "result" : "5000USD" }

3. { "msg" : "updated", "methods" : ["randomId-1"] }
```

3.5. ATMOSPHERE

Atmosphere (https://atmospherejs.com/) és un projecte de Precolate Studio i es tracta d'un catàleg d'eines, recursos i paquets per a Meteor. Des d'aquests catàleg pots explorar els paquets més populars i confiables per el teu projecte.

Qualsevol usuari que tingui un compte d'usuari de Meteor pot crear un paquet i publicar-lo en aquesta plataforma. Per aquest motiu i amb l'ajuda de la comunitat, Atmosphere inclou més de 10.000 paquets disponibles actualment.

Un cop trobat el paquet que volem incloure en el nostre projecte, ara només ens falta executar la comanda especifica a la terminal amb el nom d'usuari del desenvolupador (si el paquet és oficial de Meteor, no serà necessari) i el nom del paquet de la següent forma:

```
meteor add 'usuari':'paquet'
```

Per defecte, Metreor inclou alguns paquets en el moment de crear noves aplicacions:

- **autopublish:** (per prototips) Publica la base de dades completa a tots els clients.
- **blaze-html-templates:** Compila els templates HTML.
- **insecure:** (per prototips) Permet als clients l'escriptura a les bases de dades.
- **jquery:** Modifica el DOM (Document Object Model) utilitzant els selectors CSS.
- **meteor-base:** Paquet que totes les aplicacions meteor necessiten.
- **mobile-experience:** Paquet per millorar l'experiència en dispositius mòbils.
- **mongo:** Adaptador per utilitzar MongoDB i minimongo a traves de DDP.
- **session:** Permet utilitzar l'objecte 'Session' en el client per crear sessions i guardar dades en format clau - valor.
- **standart-minifiers:** Proveeix dos plugins 'minifiers', processen el codi per eliminar tots els caràcters innecessaris, que utilitzen les aplicacions Meteor per defecte.
- **tracker:** Permet els callbacks reactius.

De tots aquests paquets, es necessari eliminar 'autopublish' i 'insecure' per poder crear una aplicació segura i funcional, ja que aquests paquets són només per prototips de cara al desenvolupador i no al client final.

Per eliminar-los es necessari executar la següent comanda en la terminal:

```
meteor remove insecure
meteor remove autopublish
```

A continuació parlarem de varis paquets que no estan inclosos per defecte en les aplicacions de Meteor però que són de molta utilitat, es tracta de **"Iron Router"**, **"Accounts - Password"** i **"Bootstrap".**

3.5.1. Iron Router

Per instal·lar aquest paquet necessitem escriure la següent comanda per la terminal:

```
meteor add iron:router
```

Aquest paquet facilita la gestió de les URLs gràcies a les funcions de la llibreria que incorpora ('**Router**').

En els nostres projectes, questa gestió estarà situada dins la carpeta **lib/** en el fitxer anomenat **router.js** tal com està indicat en l'apartat 2.1.1 d'aquest document.

Les funcions que utilitzarem nosaltres seran bàsicament: map() i route(). Amb aquestes dues funcions és posible la gestió de totes les peticions. La sintaxi que utilitzarem serà la següent:

```
Router.map(function(){
    this.route(nom, {opcions});
});
```

On el 'nom' serà el nom del template que servirem al navegador web.

3.5.2. Accounts - Password

Per instal·lar aquest paquet necessitem escriure la següent comanda per la terminal:

```
meteor add accounts-password
```

Aquest paquet és un servei d'usuari que habilita el login segur basat en password (també inclou funcions per la recuperació del password).

Els passwords són encriptats mitjançant **bcrypt** tal com ho fan moltes companyies.

Els usuaris estan guardats en la col·lecció **'users'** de la base de dades MongoDB. El següent codi és un exemple d'usuari per observar l'estructura que segueix la base de dades:

```
{
  _id: "bbca5d6a-2156-41c4-89da-0329e8c99a4f",  // Meteor.userId()
  username: "Cool_Name", // unique name
  emails: [
    // each email address can only belong to one user.
    { address: "cool@example.com", verified: true },
    { address: "another@different.com", verified: false }
  ],
  createdAt: Wed Aug 21 2013 15:16:52 GMT-0700 (PDT),
  profile: {
    // The profile is writable by the user by default.
    name: "Joe Schmoe"
  },
  services: {
    facebook: {
      id: "709050", // facebook id
      accessToken: "AAACCgdX7G2...AbV9AZDZD"
    },
    resume: {
      loginTokens: [
        { token: "97e8c205-c7e4-47c9-9bea-8e2ccc0694cd",
          when: 1349761684048 }
      ]
    }
  }
}
```

Meteor també proporciona altres paquets similars que habiliten el login mitjançat altres serveis, com per exemple google, facebook o twitter (utilitzant els paquets accounts-google, accounts-facebook i accounts-twitter respectivament).

3.5.3. Bootstrap

Al igual que amb els altres paquests, haurem d'escriure la següent comanda per tal d'incorporar-lo en el nostre projecte:

```
meteor add twbs:bootstrap
```

Bootstrap és un front-end framework HTML, CSS i JavaScript bastant popular pel fet de que incorpora un disseny responsive i eines per gestionar fàcilment el disseny web tant per dispositius mòbils com per dispositius clàssics.

Un altre característica és el disseny atractiu que incorpora en els seus fitxers CSS.

Imatge 18: Diferents classes de botons en bootstrap

3.6. APACHE CORDOVA

Meteor integra el conjunt d'APIs **Apache Cordova** que permeten l'accés a funcions natives dels smartphones utilitzant JavaScript.

Cordova permet la creació d'apps natives per smaprtphones utilitzant HTML, CSS i JavaScript. Això permet que amb un mateix codi, puguis compilar-lo i aconseguir un arxiu '.apk' per Google Play (dispositius Android) o un arxiu '.ipa' per a Apple Store (dispositius iOS).

Per crear la nostra app nativa simplement li hem de dir a meteor que inclogui la plataforma que desitgem al projecte. Les comandes per operar amb les plataformes són les seguents:

```
meteor add-platform ios
meteor add-platform android

meteor remove-platform ios android

meteor list-platforms
```

Per tal de poder afegir la plataforma Android, primer haurem de tenir instal·lat '**Android tools**' i '**Java Development Kit**' en el nostre ordinador. La manera més facil d'aconseguir-ho és instal·lant Android Studio i descarregar-se '**SDK Platform**' de l'API 22 des de SDK Manager.

Prèviament a executar l'aplicació en el mòbil, serà necessari executar un parell de línies per configurar algunes variables:

```
export ANDROID_HOME=/home/user/Android/Sdk  //PATH per defecte en Ubuntu

export PATH=${PATH}:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools
```

Si volem executar l'aplicació en algun dispositiu, en el cas d'Android, podem utilitzar la opció '**android**' per executar-ho en un emulador o '**android-device**' per utilitzar un dispositiu conectat per USB i amb mode *'debugging'*.

```
meteor run android
meteor run android-device
```

Les comandes anteriors instal·laran una aplicació nativa en el teu dispositiu. En el cas que necessitéssim obtenir els binaris per tal d'instal·lar l'aplicació en altres dispositius (arxiu *.apk* en dispositius Android), s'ha d'executar la següent comanda:

```
meteor build "PATH" //Ha d'estar fora de la carpeta del projecte
```

Aquesta comanda ens crearà una carpeta amb tot el codi de l'aplicació, per editar-lo amb Android Studio per exemple.

Després d'executar la comanda de *build* obtindrem la nostra APK dins de ***"PATH/android/release-unsigned.apk"***. Tal com diu el nom de l'arxiu, aquest fitxer no està firmat, en el cas que vulguem instal·lar la nostra aplicació en algun dispositiu Android, necessitarem firmar-lo digitalment, en cas de no fer-ho, el SO Android no ens permetrà instal·lar aquest arxiu.

Per fer-ho hem de seguir les següents comandes:

```
//Generem una clau per la nostra app (només el primer cop que generem l'apk)
keytool -genkey -alias your-app-name -keyalg RSA -keysize 2048 -validity 10000


//Signem la app amb la clau obtinguda
cd ~/build-output-directory/android/
jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 release-unsigned.apk your-app-name


//Opcional: optimitzar la APK amb l'ajuda de zipalign
$ANDROID_HOME/build-tools/<build-tools-version>/zipalign 4 release-unsigned.apk <your-app-name>.apk
```

3.6.1. mobile-config.js

Tot i que podem executar codi exclusivament en dispositius mobils utilitzant el flag '*Meteor.isCordova*', la gran part de la configuració i les ***metadatas*** d'aquests dispositius es troba en el fitxer ***'mobile-config.js'***.

En aquest arxiu hi ha moltes configuracións possibles, aqui només ensenyaré un exemple de configuració per tal d'observar el seu aspecte i veure les variables més importants:

```
// La secció info és opcional, és on hi han definides les metadates
App.info({
  id: 'com.example.oriol.vall',
  name: 'vall',
  description: 'Get power in one button click',
  author: 'Oriol Vall Hernandez',
  email: 'contact@example.com',
  website: 'http://example.com'
});

// Definim els icones i les pantalles d'inici.
App.icons({
  'iphone': 'icons/icon-60.png',
  'iphone_2x': 'icons/icon-60@2x.png',
  // ... més mides i plataformes ...
});

App.launchScreens({
  'iphone': 'splash/Default~iphone.png',
  'iphone_2x': 'splash/Default@2x~iphone.png',
  // ...
```

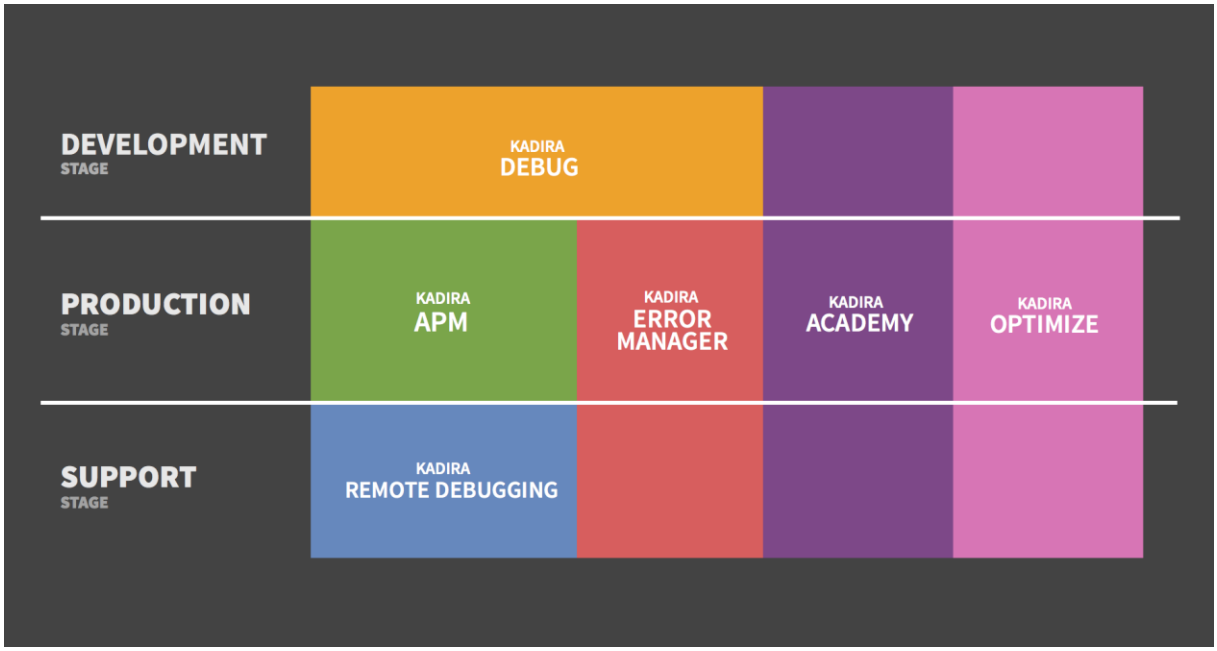
```
});

// Definim les preferencies.
App.setPreference('BackgroundColor', '0xff0000ff');
App.setPreference('HideKeyboardFormAccessoryBar', true);
App.setPreference('Orientation', 'default');
App.setPreference('Orientation', 'all', 'ios');
```

3.7. KADIRA

Imatge 19: Logotip de Kadira

Kadira és una eina que ajuda a veure que està passant amb l'aplicació Meteor en temps real durant el seu desenvolupament, producció i posterior etapa de suport i t'ho mostra en simples gràfiques per tal de poder fer un seguiment (*'tracking'*).
Per fer-ho disposem de varis serveis de Kadira:

- **Debug:** Enten que està passant dins de l'app de Meteor durant el desenvolupament.
- **APM (aplication performance monitoring):** Veu el que està passant dins de l'app en producció en temps real.
- **Error Manager:** Descobreix errors abans que els teus usuaris (tant en el client com en el servidor).
- **Academy:** Entén Meteor i aprèn a optimitzar-lo.



Imatge 20: Esquema dels diferents serveis de Kadira en funció de l'etapa actual de l'app

Per tal d'enllaçar la nostra app amb Kadira és necessari intal·lar dos paquets:

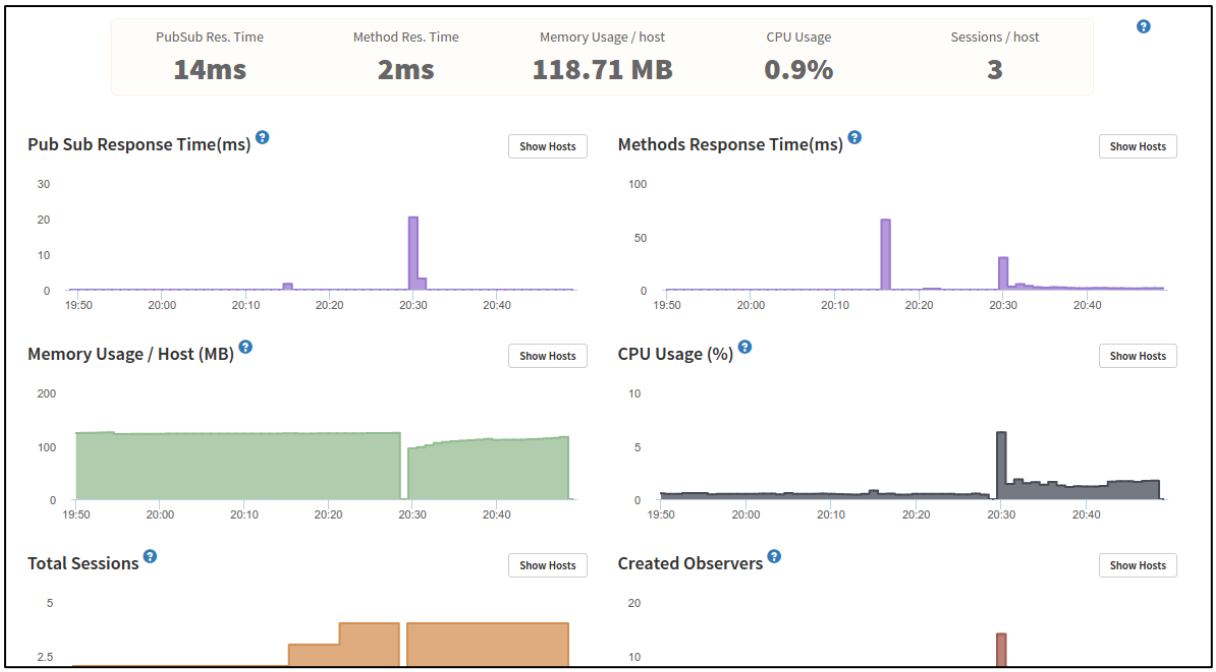
```
meteor add meteorhacks:kadira
meteor add kadira:debug
```

I aleshores hem de crear l'aplicació a Kadira per tal d'obtenir les credencials per utilitzar l'API i guardar-les dins de **server/kadira.js**

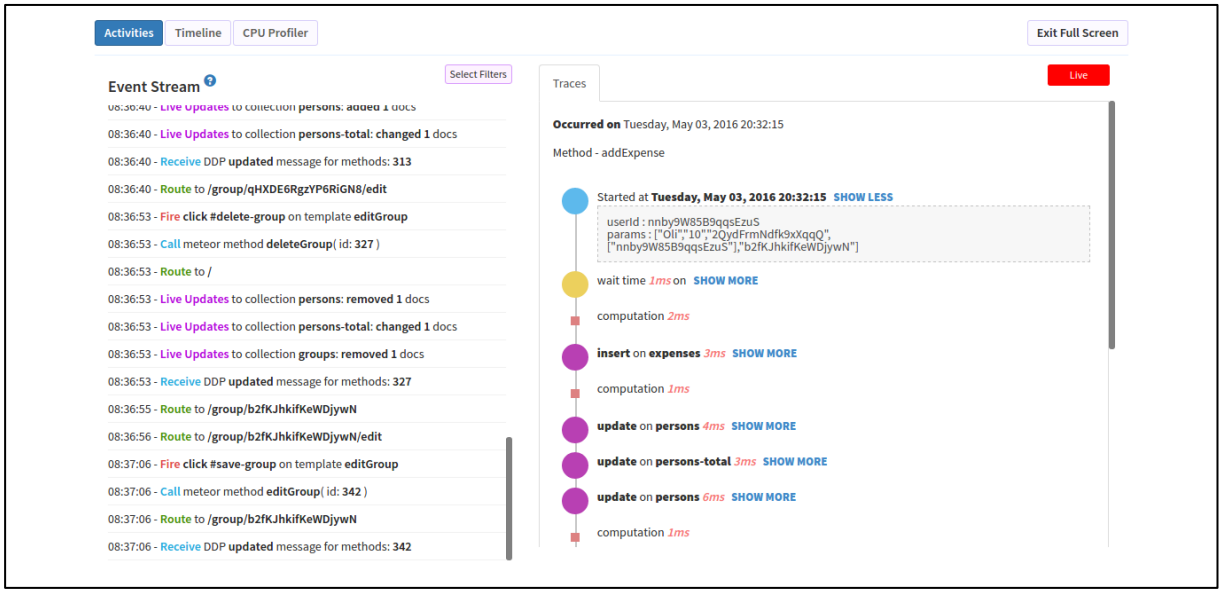
Un exemple de fitxer **kadira.js** seria aquest:

```
Kadira.connect('C4hqD5MZhnjRkPCAd', 'b69374ef-6c6d-407e-be0f-bddf1e3da854');
```

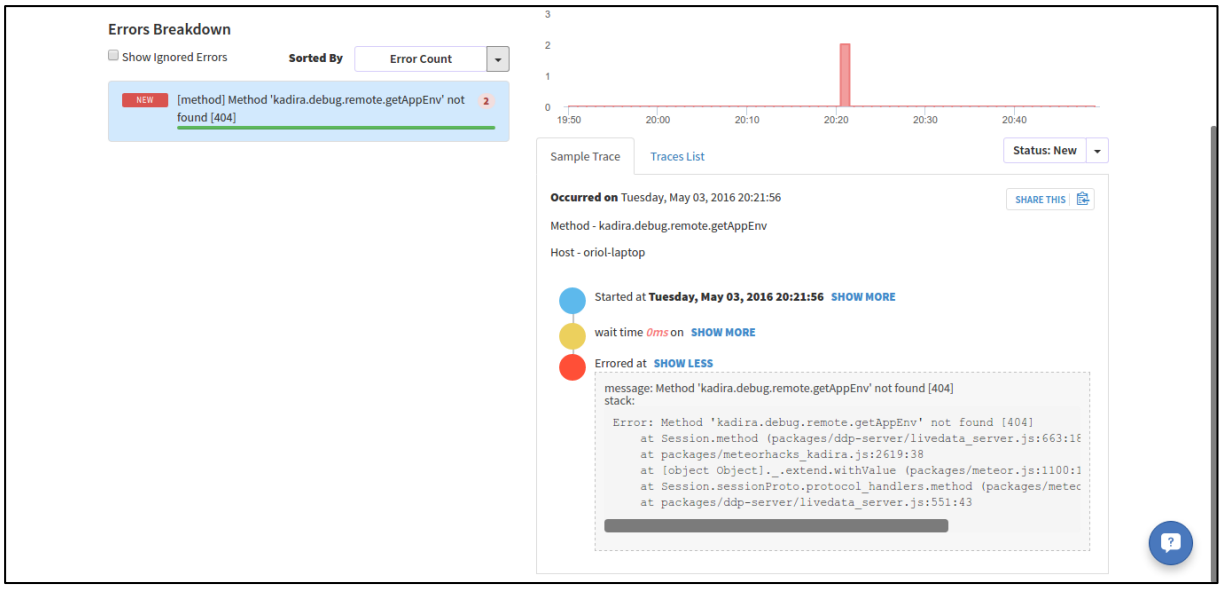
A partir d'aquest moment ja podriem accedir a http://debug.kadiraio.com/debug (per DEBUG) i a https://ui.kadira.io (per APM i ERROR MANAGER).



Imatge 21: Exemple Kadira APM



Imatge 22: Exemple Kadira Debug



Imatge 23: Exemple Kadira Error Manager

3.8. HEROKU

Imatge 24: Logotip Heroku

Per tal de *desplegar (‘deploy’)* les nostres apps hi han varies plataformes que permeten gestionar petits servidors online (cloud servers) i et proporcionen un domini de forma gratuïta. Alguns exemples són: Heroku, Rackspace, Amazon AWS, OpenShift, Linode o DigitalOcean.

Per tal de provar les apps creades per aquest projecte jo vaig decidir a utilitzar Heroku a causa de les seves funcionalitats per allotjar aplicacions Meteor:

- Un **subdomini** per a cada aplicació dins del domini 'heroku.com'.
- Molta **facilitat** alhora de desplegar aplicacions.
- Servidors disponibles en diferents **llenguatges** (Node.js en el nostre cas)
- **Add-ons** per afegir funcionalitats extra (com per exemple una BD de MongoDB).
- **Buildpacks** de diferents llenguatges per automatitzar la creació de builds (buildpack de Meteor per un servidor Node.js per exemple).
- Opció per desenvolupar varies apps de manera **gratuïta**.

Per tal de poder utilitzar Heroku és necessari tenir instal·lat el ***toolbelt.***
La comanda per sistemes Ubuntu/Debian és la següent:

```
wget -O- https://toolbelt.heroku.com/install-ubuntu.sh | sh
```

Quan tinguem instal·lat el toolbelt, podrem utilitzar la comanda '**heroku**' per tal de gestionar les nostres aplicacións.

Per tal d'executar les nostres aplicacions a Heroku necessitarem configurar-les (assignar un buildpack i una base de dades) amb les següents comandes:

```
heroku buildpacks:set https://github.com/AdmitHub/meteor-buildpack-horse.git

heroku addons:create mongolab
```

4. APLICACIONS DESENVOLUPADES

Per demostrar el gran potencial d'aquesta eina s'han desenvolupat varies aplicacions d'exemple per aquest treball utilitzant les eines comentades en aquest treball.

4.1. LOGIN BÀSIC

Aquesta aplicació és simplement una pàgina d'inici amb un *'login'* i una pàgina per registrar-se en cas de no tenir cap compte d'usuari. Un cop s'ha iniciat la sessió, l'usuari pot accedir a la pàgina on hi han les dades.

Les altres aplicacions que també necessiten un registre d'usuaris han utilitzat aquesta com a plantilla.

En aquesta aplicació hem utilitzat els paquets "**iron:router**" i "**accounts-password**". Per instal·lar-los s'ha d'executar les següents comandes en el directori arrel de l'aplicació:

```
meteor add iron:router
meteor add accounts-password
```

En aquest projecte no serà necessària la carpeta 'server' ja que no hi haurà dades en el servidor, només a MongoDB (administrat pel paquet 'accounts-password'), per tant l'estructura final serà la següent:

```
login/
├── client
│   ├── controllers
│   │   └── usersAccounts.js
│   └── views
│       ├── home.html
│       ├── layout.html
│       └── register.html
└── lib
    └── router.js
```

Dins de l'arxiu **'router.js'** (encarregat de la gestió de les URLs) només hi ha dues URLs:

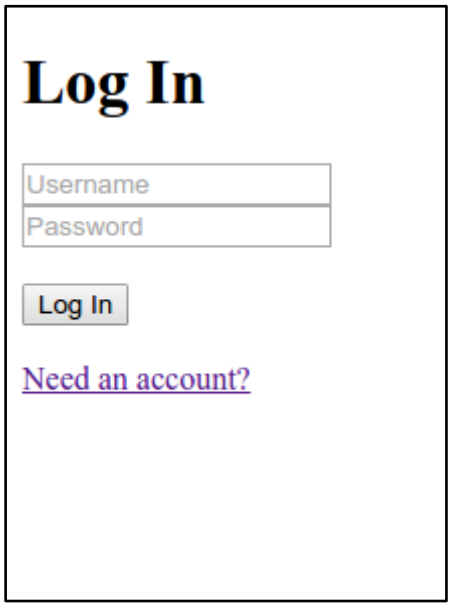
- **'/'** : Adreça arrel del projecte, on el contingut només serà disponible per usuaris registrats.
- **'/register'** : Adreça amb un formulari per registrar-se.

En el fitxer **home.html** utilitzem els helpers **'currentUser'** i **'loggingIn'** que són proporcionats pel paquet **'accounts-password'** i que ens indiquen si el client ha iniciat sessió o està iniciant la sessió en aquell moment.

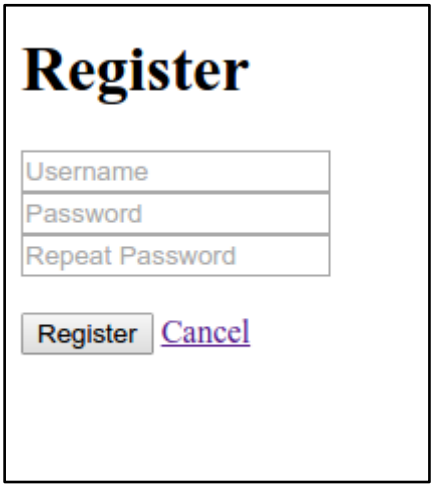
Un altre funció de la mateixa llibreria que utilitzarem serà : **createUser()**. En aquesta funció li passem com a paràmetres un objecte JSON amb el nom d'usuari i el password i una funció callback que s'executarà després de crear l'usuari i comprovarà si hi ha algun error.

A part de crear l'usuari, aquesta funció ens comprova que l'usuari no estigui registrat encara, encripta el password abans de que arribi al servidor i inicia la sessió després del registre.

Per tal d'iniciar i tancar sessions, utilitzem les funcions **loginWithPassword()** i **logout()** (inclosos també amb el paquet accounts-password).



Imatge 25: Pantalla d'inici de l'aplicació Login

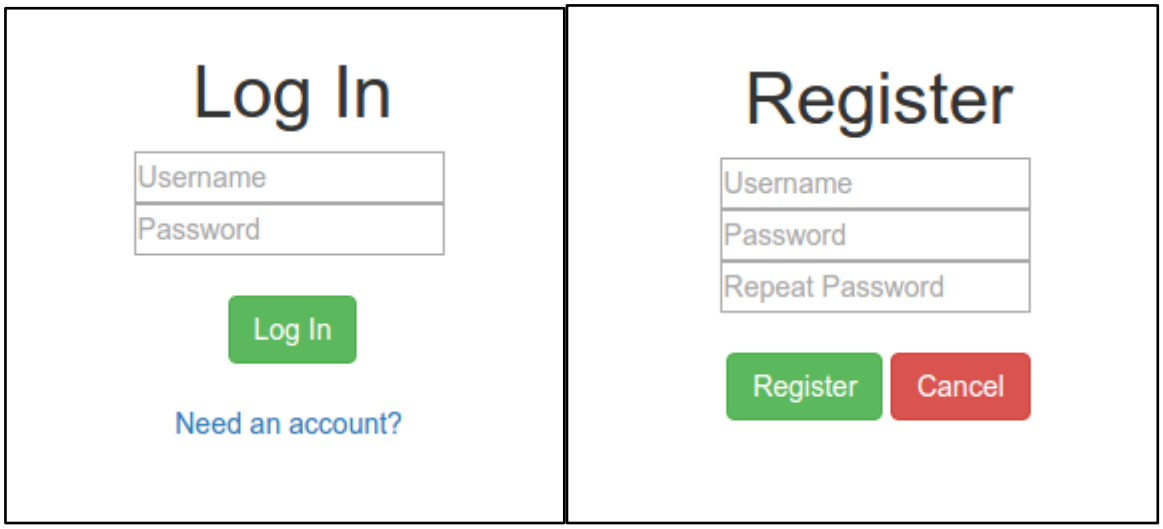


Imatge 26: Pantalla de registre

Imatge 27: Pantalla d'inici un cop iniciada la sessió

Per tal de fer la nostra aplicació *responsible* hem afegit el paquet de bootstrap i així també hem pogut canviar l'aspecte de la nostra aplicació.

Una característica important dels fitxers **html** és l'etiqueta **'<meta>'**, aquesta línia serveix per crear una vista ampliada amb els dispositius mòbils i així facilitant el seu ús en aquests dispositius.



Imatge 28: Pantalles de Log In i Register amb Bootstrap

4.2. JOC MULTIJUGADOR EN TEMPS REAL

Aquesta aplicació consistirà amb un joc estil Trivial (preguntes amb quatre respostes possibles) en el qual hi hauran partides de màxim 4 jugadors en temps real (per demostrar la funcionalitat del temps real, tots els jugadors respondran les preguntes al mateix temps).

Per la seva estructura, aquest es un projecte bastant interessant per mostrar les capacitats de Meteor ja que el fet de ser multijugador i temps real aporten certa dificultat que amb altres plataformes seria bastant complicat d'aconseguir.

Utilitzarem el sistema Login de l'apartat anterior com a plantilla.

L'estructura resultant d'aquesta aplicació és la següent:

```
logic-game/
├── client
│   ├── controllers
│   │   ├── menu.js
│   │   ├── roomsControl.js
│   │   └── usersAccounts.js
│   ├── helpers
│   │   ├── roomHelper.js
│   │   └── username.js
│   └── views
│       ├── home.html
│       ├── layout.html
│       ├── menu.html
│       ├── register.html
│       └── rooms.html
├── lib
│   └── router.js
└── models
    ├── gameCreation.js
    ├── games.js
    ├── questions.js
    ├── roomCreation.js
    └── rooms.js
```

A part de la col·lecció a MongoDB per els usuaris, és necessari crear tres col·leccions més per aquesta aplicació:

- *Rooms:* Encarregada de registrar les partides que s'estan duent a terme.
- *Questions*: Base de dades amb les preguntes i possibles respostes (indicant la resposta correcte).
- *Games:* Col·lecció per gestionar l'estat del joc dins d'una 'room'.

A continuació hi ha un exemple d'un document de cada col·lecció per observar el seu aspecte:

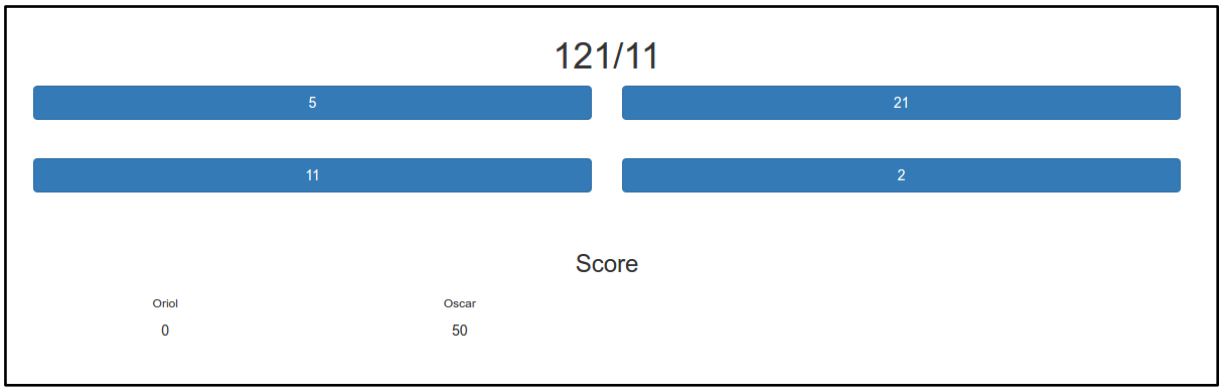
```
room = {
    private : false,
    players : [],
    playersCount: 0,
    playing: false,
    displayingResults: false,
    full: false,
    game: GameObject
  }

game = {
    questions : QuestionsArray,
    players : [{player : playerId, score : 0},{player : player2Id, score : 0}]
  }

question = {
    question: "2+2",
    correctAnswer: "4",
    otherAnswers: ["2","0","6"]
  }
```

En el moment en que un jugador crea una partida nova, un objecte de la classe 'Room' serà creat, i aleshores, en el moment de començar la partida, crearem un objecte 'Game' amb l'estat de la partida (marcadors, i les preguntes escollides aleatoriament per aquella partida).

L'objecte **'room'** té la propietat '*private*' per tal de definir si aquella room pot ser trobada per altres usuaris (mitjançant el botó '*play now*') o necessites la seva ID per trobar-la utilitzant la pantalla de '*Search Game*' (les ID es creen automaticament per la base de dades de Meteor).

Imatge 29: Botons de la pantalla d'inici

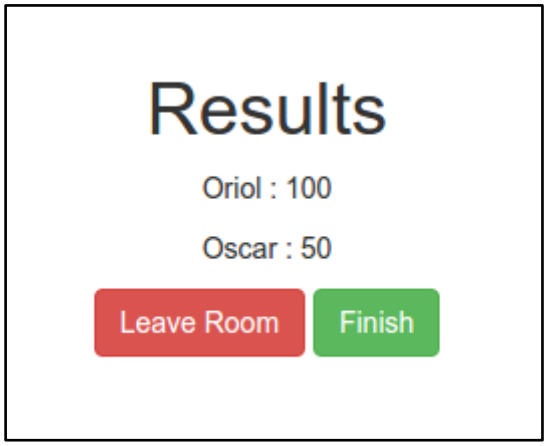
Per tal de poder començar una partida, es necessària la presencia d'almenys 2 jugadors dins de la room, aleshores, abans de començar la partida, s'assignaran les preguntes aleatoriament de la base de dades i es posaran les respostes en un ordre aleatori.



Imatge 30: Exemple de pantalla d'una pregunta

Les preguntes es respondran en temps real per tots els jugadors, cada pregunta resposta correctament sumarà 50 punts al marcador, en canvi, cada pregunta resposta de forma incorrecte restarà 75 punts. Aquesta diferencia de punts evitarà intentar respondre les preguntes al atzar.

Després de respondre totes les preguntes apareixerà la pantalla amb els resultats finals.



Imatge 31: Pantalla final amb els resultats de la partida

4.3. GESTOR DE DESPESES PER GRUPS

Aquesta aplicació serveix per compartir despeses entre usuaris. El funcionament és similar al d'altres aplicacions dissenyades per a dispositius mòbils (*Splitwise* o *Venmo* per exemple).

Imatge 32: Logotip Money App

Les despeses s'organitzen en grups, així es pot gestionar el balanç entre els membres d'un viatge o d'un grup de persones específiques. El creador del grup és l'única persona que pot editar-lo (canviar el nom i afegir persones), la resta de membres només poden escollir deixar el grup (en el cas que el seu balanç en grup sigui de 0).

Per aquest projecte he utilitzat una llibreria per millorar l'aspecte dels pop-ups i alertes anomenada "*Sweet Alert*". Amb aquesta llibreria és possible mostrar missatges d'error o alertes per assegurar les operacions d'esborrar.
Per utilitzar les funcionalitats d'aquesta llibreria s'ha de cridar la funció per executar les alertes (***swal()***) amb els paràmetres necessaris en cada cas.

Imatge 33: Logotip de la llibreria JavaScript '*Sweet Alert'*

L'estructura de fitxers d'aquesta aplicació en la seva versió final és la següent:

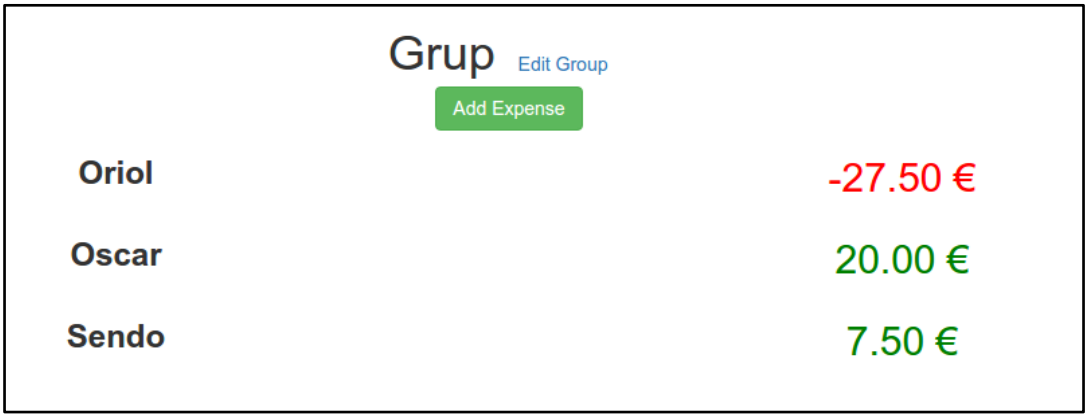
```
.
├── client
│   ├── controllers
│   │   ├── expensesController.js
│   │   ├── groupsController.js
│   │   ├── infoController.js
│   │   ├── mainController.js
│   │   ├── searchController.js
│   │   └── usersAccounts.js
│   ├── helpers
│   │   ├── helper.js
│   │   └── registerHelpers.js
│   └── views
│       ├── expenses.html
│       ├── groups.html
│       ├── home.html
│       ├── info.html
│       ├── layout.html
│       ├── main.html
│       ├── register.html
│       └── search.html
├── lib
│   └── router.js
├── mobile-config.js
├── models
│   ├── debts.js
│   ├── expenses.js
│   ├── groups.js
│   └── persons.js
├── public
```

```
│   └── images
│       └── logo.png
├── res
│   ├── mipmap-hdpi
│   │   ├── ic_launcher.png
│   │   └── icon_app.png
│   ├── mipmap-mdpi
│   │   ├── ic_launcher.png
│   │   └── icon_app.png
│   └── mipmap-xhdpi
│       ├── ic_launcher.png
│       └── icon_app.png
└── server
    └── kadira.js
```

En referència a la base de dades, aquesta aplicació consta de 5 col·leccións (a part de la col·lecció d'usuaris):

- **Persons**: Són usuaris dins d'un grup, mantenen el balanç d'aquell grup.
- **Groups**: Els documents d'aquesta col·lecció tenen un registre dels seus membres ('Persons') actuals i anteriors per si s'han de recuperar.
- **Expenses**: Cada despesa és registrada en aquesta col·lecció.
- **Debts**: Manté un registre de totes les despeses d'un mateix usuari.
- **PersonsTotal**: S'utilitza per registrar el balanç total d'un usuari.

Després de fer login en aquesta aplicació ens trobem amb la llista de grups els quals sóm membres i un botó per tal de crear nous grups. Dins de cada grup podem veure el balanç de tots els membres del grup i una llista de totes les despeses.



Imatge 34: Exemple de pantalla amb el balanç de cada usuari d'un grup

En aquesta aplicació també tenim la opció de veure el nostre perfil. En aquesta pantalla hi trobarem totes les despeses on hi participem (ja sigui com a perquè la despesa l'hem pagat nosaltres o perquè hem de pagar a algú).

En cas de no tenir un balanç de zero, en el mateix perfil ens apareixerà a qui l'hi hem de pagar o de quin grup ens han de pagar.

Imatge 35: Exemple pantalla de perfil

En el cas de deure diners a algú, podrem veure un botó de '**Settle up**' per tal d'afegir una despesa automàticament per dir que hem pagat el deute que teniem.

4.4. APP PER ORGANITZAR COLLES CASTELLERES

Al principi aquesta app era per dissenyar les pinyes dels castells de forma automàtica, però hi havien masses variables per automatitzar-ho.
Per aquest motiu vaig decidir fer una aplicació per tal de mantenir un registre d'assistència en els esdeveniments que organitza la colla (assaigs i actuacions per exemple) i a més poder muntar una pinya amb les persones que han confirmat l'assistència.

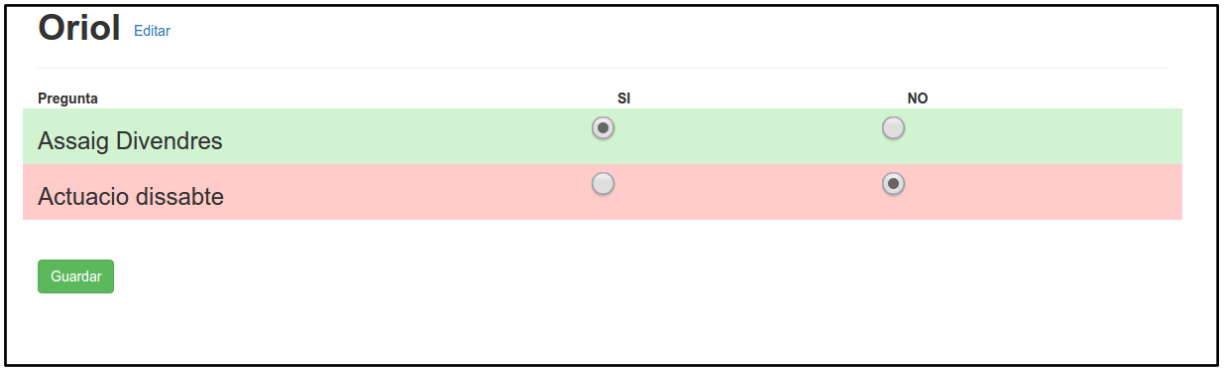
Per aconseguir-ho, vaig dividir l'aplicació en dos rols: **Cap de Colla** i **Casteller**.

Els Castellers podran veure els esdeveniments de la seva colla i podran contestar dient si assistiran o no. En canvi, els cap de colla, podran crear nous esdeveniments, veure qui ha confirmat l'assistència i crear pinyes.

```
.
├── client
│   ├── controllers
│   │   ├── accountsController.js
│   │   ├── capdeController.js
│   │   ├── castellersController.js
│   │   ├── pinyaController.js
│   │   └── preguntesController.js
│   ├── css
│   │   └── fonts.css
│   ├── helpers
│   │   ├── helpers.js
│   │   ├── registerHelpers.js
│   │   └── sAlert.js
│   ├── sidebar
│   │   └── simple-sidebar.css
│   └── views
│       ├── capdecolla.html
│       ├── castellers.html
│       ├── chart.html
│       ├── layout.html
│       ├── main.html
│       ├── pinya.html
│       ├── preguntes.html
│       └── register.html
├── config.push.json
├── lib
│   └── router.js
├── models
│   ├── colles.js
│   ├── preguntes.js
│   └── users.js
└── server
    └── startup.js
```

A la base de dades, hi podem trobar 3 taules: "users", "colles" i "preguntes".

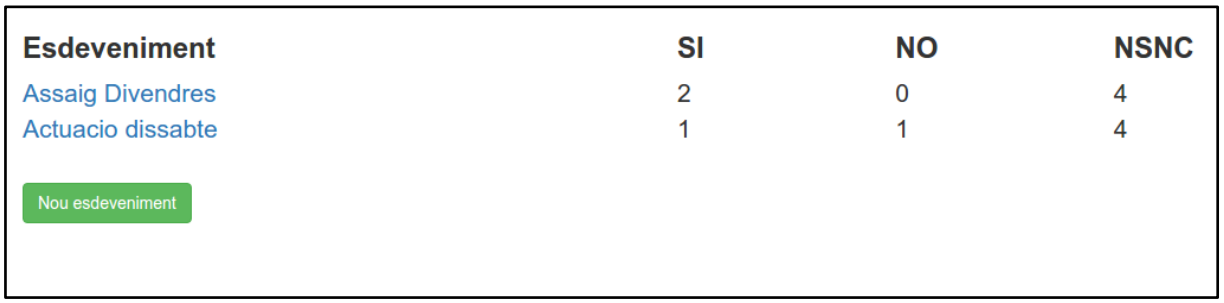
En aquesta aplicació s'ha implementat una barra lateral de navegació en la vista de cap de colla. Per implementar aquesta barra es va utilitzar una plantilla extreta de *startbootstrap.com* (aquesta plantilla és bàsicament un arxiu **css**).



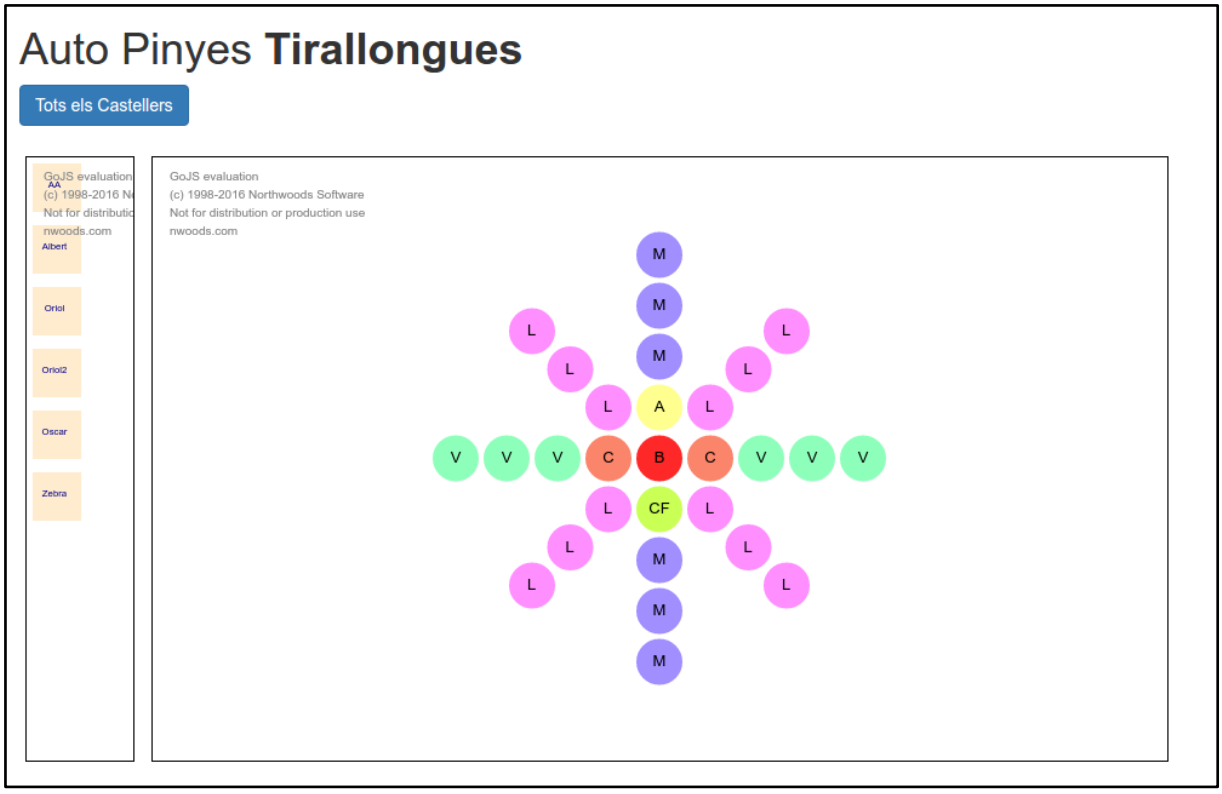
Imatge 36: Pantalla per castellers

Si no tens el rol de cap de colla, l'únic que pots veure és la pantalla amb les preguntes i dir si assistiràs o no i la pantalla per editar el teu nom, alçada i posicions preferides.

En cas de ser cap de colla, es podrà accedir a tres pantalles més: "Castellers", "Esdeveniments" i "Pinya".
A la pantalla de "Castellers" es pot trobar una llista amb tots els usuaris que pertanyen a la teva colla per poder editar-los. A la pantalla de preguntes es veuen totes les preguntes formulades, més les respostes dels castellers. Finalment, la pàgina de "Pinyes" és una plantilla d'una pinya de pilar per tal de col·locar els castellers a sobre.



Imatge 37: Pantalla "Esdeveniments"



Imatge 38: Pantalla “Pinya”

És possible limitar els usuaris disponibles a la pantalla de Pinyes accedint a la informació d’una pregunta, així, filtraríem els castellers, col·locant només els que han dit que si.

El diagrama de la pinya del pilar ha estat dissenyat utilitzant una versió d’avaluació de la llibreria **GoJS**, llibreria que permet fer diagrames i gràfiques dinàmiques. D’aquesta llibreria, s’ha utilitzat el diagrama per organitzar taules i cadires amb la diferència que només hi han “cadires” simbolitzant els llocs de la pinya.

4.5. QUALITY ENGINE PLATFORM

Aquesta aplicació ha estat desenvolupada dins el departament de qualitat de software de l’empresa IDNEO. Tot i que va començant sent una aplicació, actualment podem classificar aquest exemple com a *plataforma* degut a que ofereix diversos serveis (Tollgate Evaluation, Checklist Manager, Offers + KPIs Database i Timelines). Bàsicament és un lloc on tots els

membres de l'empresa poden veure l'estat dels projectes i ofertes de forma oberta i en temps real.

L'estructura de fitxers final és molt extensa degut a totes les funcions que inclou aquesta plataforma.

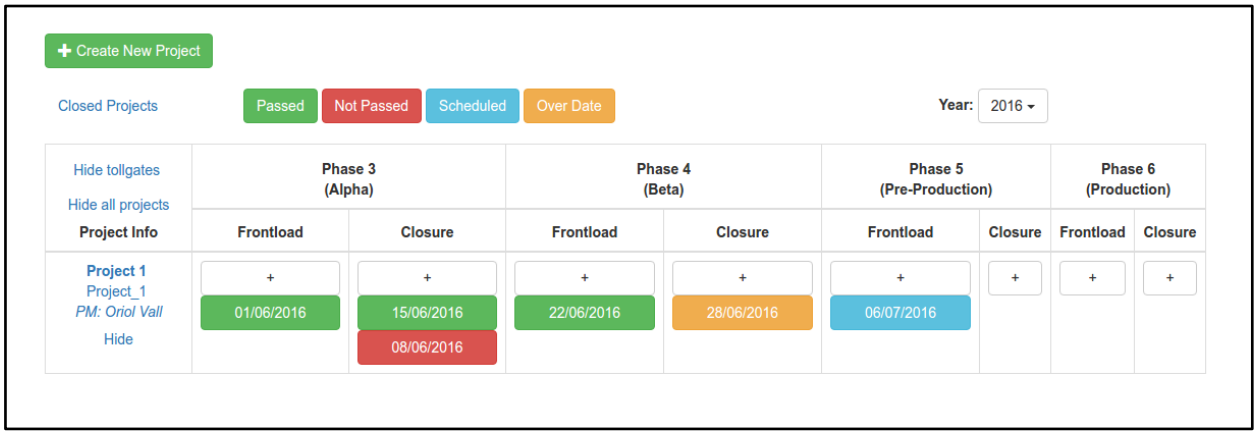
```
.
├── client
│   ├── controllers
│   │   ├── categoriesController.js
│   │   ├── chartController.js
│   │   ├── checklistsController.js
│   │   ├── checksController.js
│   │   ├── eventsController.js
│   │   ├── homeController.js
│   │   ├── kpiController.js
│   │   ├── offersController.js
│   │   ├── phasesController.js
│   │   ├── projectsController.js
│   │   ├── releaseControler.js
│   │   ├── releaseTimeController.js
│   │   ├── schemesController.js
│   │   ├── timelineController.js
│   │   ├── tollgatesController.js
│   │   ├── typesController.js
│   │   ├── usersController.js
│   │   └── yearController.js
│   ├── helpers
│   │   ├── registerHelpers.js
│   │   ├── V1Helpers.js
│   │   ├── V2Helpers.js
│   │   ├── V3Helpers.js
│   │   └── V4Helpers.js
│   └── views
│       ├── categories.html
│       ├── chart.html
│       ├── checklists.html
│       ├── checks.html
│       ├── css
│       │   └── css.css
│       ├── events.html
│       ├── home.html
│       ├── index.html
│       ├── KPIs.html
│       ├── layout.html
│       ├── offers.html
│       ├── phases.html
│       ├── projects.html
│       ├── releases.html
│       ├── releaseTime.html
```

```
│       ├── shemes.html
│       ├── timeline.html
│       ├── tollgates.html
│       ├── types.html
│       └── users.html
├── lib
│   └── router.js
├── logs.txt
├── models
│   ├── checklists.js
│   ├── checksGroups.js
│   ├── checks.js
│   ├── eventsTime.js
│   ├── KPIs.js
│   ├── logs.js
│   ├── offers.js
│   ├── phaseReleases.js
│   ├── phasesTime.js
│   ├── projects.js
│   ├── releases.js
│   ├── releasesTime.js
│   ├── schemes.js
│   ├── targets.js
│   ├── tollgates.js
│   ├── types.js
│   ├── users.js
│   └── years.js
├── package.json
├── public
│   └── images
│       ├── logo_idneo_favicon.png
│       └── logo_idneo.png
└── server
    └── startup.js
```

Al principi, l'objectiu era traslladar les dades d'una fulla excel a una aplicació web (imitant el sistema d'una aplicació similar creada en *php* anomenada SPI) per tal de poder-la modificar i veure-la en temps real des d'un navegador.

La primera versió de l'aplicació només gestionava les dates i l'estat final dels **Tollgates** (esdeveniments específics per determinar si un projecte va per bon camí). Per tant, l'aplicació constava simplement d'una taula de projectes amb columnes que significaven les fase dels projectes. Aleshores un administrador podia crear nous tollgates dins d'un projecte especificant la data. Cada tollgate té un estat (Scheduled, Passed, Not Passed, Undefined) i un link a SVN (opcional) amb la Checklist corresponent per demostrar l'estat del tollgate.

Per tal d'aconseguir l'aplicació es van crear les següents taules a MongoDB: 'projects', 'releases', 'years' i 'tollgates'.

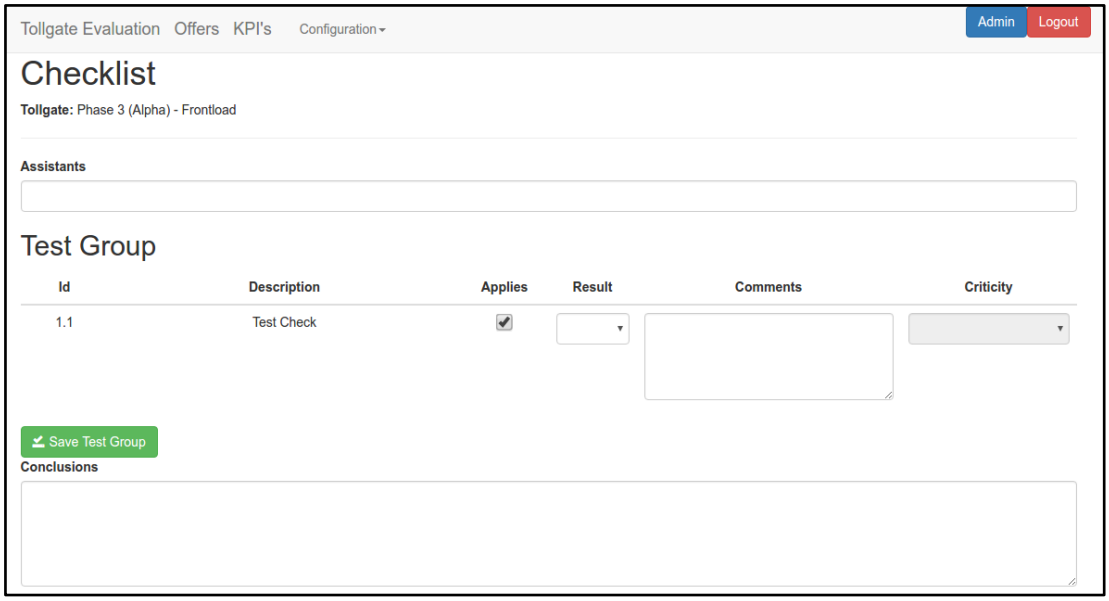


Imatge 39: Pantalla principal de Tollgate Evaluation

Un cop acabada la primera versió, es va ampliar per tant de poder gestionar les checklist des de la mateixa aplicació. Per tant es va ampliar afegint les taules 'checklists', 'checks', 'checksGroups', 'types' i 'schemes'.

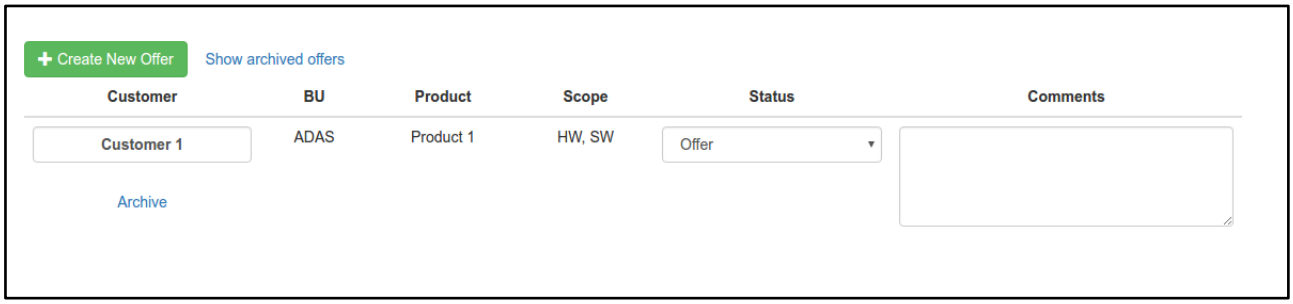
Per tal de fer un sistema escalable, vaig inspirarme en el sistema que utilitza JIRA per tal de gestionar els seus *Workflows* i *Issues*. El sistema consta en l'ús de plantilles i esquemes. Cada projecte té associat un esquema que és un conjunt de plantilles de Checklists, per tant, cada cop que es crea un tollgate, es crea una còpia de la plantilla del checklist, d'aquesta manera pots modificar les plantilles en qualsevol moment, i els canvis apareixeran en les futures checklists.

Cada Check consta d'un identificador, una descripció, un resultat seguit d'un camp de criticalitat (en cas de tenir un resultat negatiu), un camp de comentaris i una checkbox per definir si el check és aplicable en aquest projecte.

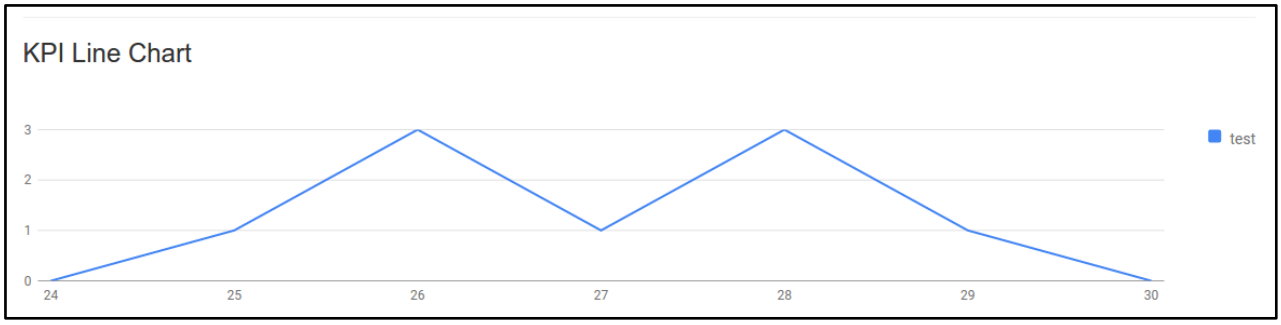


Imatge 40: Pàgina de Checklist amb un check d'exemple

La següent versió de la plataforma va incloure una gestió d'ofertes i dades per mesurar la qualitat del software (KPIs). Fins aquell moment, aquestes dades s'anaven emmagatzemant en fulls d'excel. La plataforma va canviar això guardant les dades en una base de dades i permeten la visualització online. A més, amb la utilització de la llibreria de Google: **Google Charts** vaig implementar unes gràfiques linieals amb les dades dels KPIs.



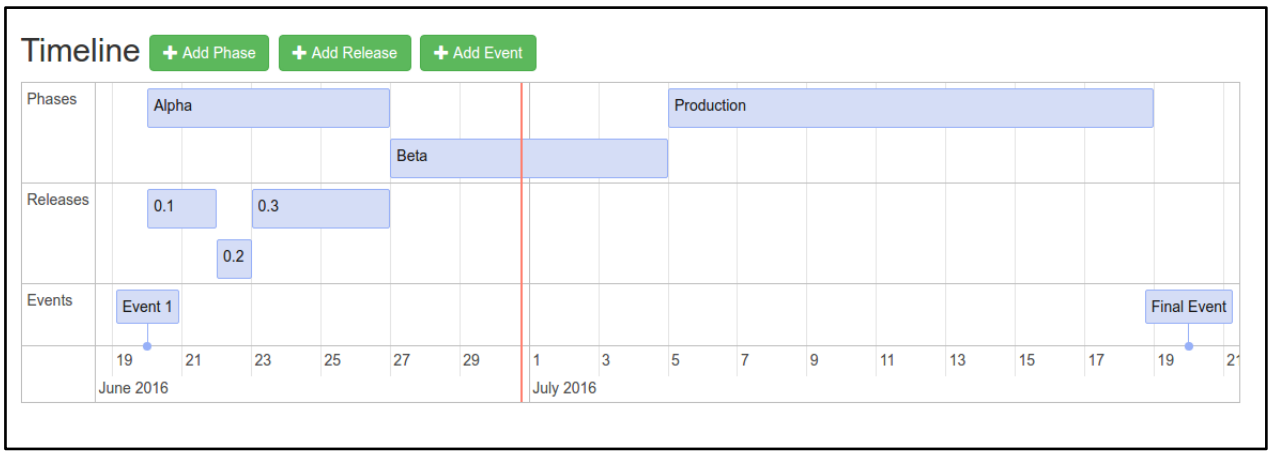
Imatge 41: Exemple d'una oferta



Imatge 42: Exemple gràfica de Google Chart

Per tal de gestionar aquestes dades es van crear més taules a la base de dades ('targets', 'offers' i 'kpis').

L'última implementació feta en aquesta plataforma fins el moment és la incorporació d'una **Timeline** dels projectes. Per tal de mostrar una vista global del projecte amb totes les seves fases, releases i esdeveniments, d'aquesta manera es pot mostrar l'estat del projecte amb detall i més personalitzats que amb la taula de Tollgates.
Per implementar la timeline es va utilitzar una llibreria de javascript anomenada **visjs** que incorpora la possibilitat de crear varies gràfiques (la timeline entre elles).



Imatge 43: Exemple de Timeline amb fases, releases i esdeveniments

Actualment, aquesta plataforma segueix en desenvolupament. Els següents passos són els de sincronitzar la taula de Tollgates amb la Timeline, modificar els colors dels objectes de la timeline per mostrar el seu resultat i treballar amb la API de JIRA per tal de consultar dades i fer la gestió de KPIs més automàtica.

5. CONCLUSIONS

Hem comprovat que els frameworks web són una ajuda molt útil en el desenvolupament d'aplicacions web i pel que es pot deduir, aquestes eines aniran implementant cada cop més funcionalitats amb el temps.

Entre les diverses opcions, aquest projecte ha estat centrat en l'anomenat Metor per les seves característiques de real-time i poder programar utilitzant el mateix llenguatge tant al client com al servidor. Hem investigat perquè aquestes característiques són possibles en Meteor (gràcies al protocol DDP i al sistema de subscripcións de Mongo més l'ús de Node.js per part del servidor) i no en altres frameworks. També s'ha demostrat com podem millorar les aplicacións creades amb Meteor gràcies als paquets que ofereix Atmosphere i les dades que proporciona Kadira. Per acabar, també s'ha comentat eines per tal de desplegar les aplicacións online, sense haver d'asumir el cost de la infrestructura, grácies a Heroku.

Durant la segona part del projecte, dedicada a crear aplicacions d'exemple, hem vist aplicats tots els conceptes apresos anteriors junt amb l'ús de noves llibreries JavaScript (*visjs* per la Timeline en l'aplicació de Quality Engine, per exemple, o llibreries per crear alertes més visuals, *SweetAlert*). Durant el procés de creació van aparèixer problemes, com per exemple poder crear una gràfica dinàmica per fer pinyes en l'aplicació de castellers o aconseguir mesurar la manera més simple de pagar els deutes a l'aplicació de gestió de despeses. Peró, finalment, han sortit bones aplicacions.

Com a conclusió, personalment crec que Meteor és un bon framework tot i les mancances actuals (falta de suport per bases de dades SQL, entre d'altres), que segurament es solucionaran en un futur pròxim. Peró per tenir una visió més objectiva crec que hauria de treballar amb un altre framework, ja que no tinc altre referència a part de comentaris en pàgines web. Per solucionar-ho hauríem d'estudiar i treballar amb altres frameworks (Django o web2py semblen les millors alternatives) i provar de fer algunes aplicacions per tal de comparar la facilitat de creació durant el desenvolupament i les sensacions de l'usuari final. Només llavors podrem decidir si hi ha un framework millor que els altres o si cada framework té els seus advantatges en funció de l'aplicació que es vulgui desenvolupar.

6. BIBLIOGRAFIA

[1]"Meteor", *Meteor.com*, 2016. [Online]. Available: https://www.meteor.com/. [Accessed: 11- Feb- 2016].

[2]"What is Meteor.js?", *Joshowens.me*, 2014. [Online]. Available: http://joshowens.me/what-is-meteor-js/. [Accessed: 30- Feb- 2016].

[3]"Meteor (web framework)", *Wikipedia*, 2016. [Online]. Available: https://en.wikipedia.org/wiki/Meteor_(web_framework). [Accessed: 17- Feb- 2016].

[4]"Introducción a Node.js - Rafa Muñoz", *Rmunoz.net*, 2011. [Online]. Available:http://www.rmunoz.net/introduccion-a-node-js.html. [Accessed: 05- Apr- 2016].

[5]"Handlebars.js: Minimal Templating on Steroids", *Handlebarsjs.com*, 2016. [Online]. Available: http://handlebarsjs.com/. [Accessed: 30- Mar- 2016].

[6]"Understanding Meteor Internals | Pro Meteor", *Meteorhacks.com*, 2016. [Online]. Available: https://meteorhacks.com/understanding-meteor-internals/. [Accessed: 20- Feb- 2016].

[7]"Descubriendo Meteor", *Es.discovermeteor.com*, 2016. [Online]. Available: http://es.discovermeteor.com/. [Accessed: 29- Mar- 2016].

[8]"Deploy to production on Heroku - Just Meteor", *Just Meteor*, 2015. [Online]. Available: http://justmeteor.com/blog/deploy-to-production-on-heroku/. [Accessed: 26- Apr- 2016].

[9]"Introducing Meteor API Docs | Meteor API Docs", *Docs.meteor.com*, 2016. [Online]. Available: http://docs.meteor.com/. [Accessed: 15- Feb- 2016].

[10]"Introduction | Meteor Guide", *Guide.meteor.com*, 2016. [Online]. Available: http://guide.meteor.com/. [Accessed: 17- Feb- 2016].

[11]"Slant - What are the best full-stack isomorphic JavaScript frameworks?", *Slant*, 2016. [Online]. Available:http://www.slant.co/topics/3918/~full-stack-isomorphic-javascript-frameworks. [Accessed: 29- Apr- 2016].

[12]"angular-meteor - realtime full stack", *Angular-meteor.com*, 2016. [Online]. Available: http://www.angular-meteor.com/. [Accessed: 09- May- 2016].

[13]"Django vs Meteor vs Rails Stackup | StackShare", *StackShare*, 2016. [Online]. Available:http://stackshare.io/stackups/rails-vs-django-vs-meteor. [Accessed: 02- Jun- 2016].

[14]"meteorjs vs. Django vs. web2py comparison | vsChart.com", *Vschart.com*, 2016. [Online]. Available:http://vschart.com/compare/meteorjs/vs/django-framework/vs/web2py. [Accessed: 09- Jun- 2016].

[15]"Python Frameworks: Full-Stack vs. Micro Framework - DZone Web Dev", *dzone.com*, 2016. [Online]. Available:https://dzone.com/articles/python-frameworks-full-stack-vs-micro-framework. [Accessed: 01- Jun- 2016].

[16]"The Web framework for perfectionists with deadlines | Django", *Djangoproject.com*, 2016. [Online]. Available: https://www.djangoproject.com/. [Accessed: 06- Jun- 2016].

[17]"Cloud Application Platform | Heroku", *Heroku.com*, 2016. [Online]. Available: https://www.heroku.com/. [Accessed: 03- Apr- 2016].

[18]"Rails Hosts: Amazon AWS vs. Digital Ocean vs. Heroku vs. Engine Yard", *Airpair.com*, 2016. [Online]. Available:https://www.airpair.com/ruby-on-rails/posts/rails-host-comparison-aws-digitalocean-heroku-engineyard. [Accessed: 13- Apr- 2016].

[19]"Kadira - Performance Monitoring Platform for Meteor", *Kadira.io*, 2016. [Online]. Available: https://kadira.io/. [Accessed: 21- May- 2016].

[20]"Apache Cordova", *Cordova.apache.org*, 2016. [Online]. Available: https://cordova.apache.org/. [Accessed: 12- Apr- 2016].

[21]"Bootstrap · The world's most popular mobile-first and responsive front-end framework.", *Getbootstrap.com*, 2016. [Online]. Available: http://getbootstrap.com/. [Accessed: 09- Apr- 2016].

[22]"The trusted source for JavaScript packages, Meteor resources and tools | Atmosphere",*Atmospherejs.com*, 2016. [Online]. Available: https://atmospherejs.com/. [Accessed: 15- Apr- 2016].

[23]"The MongoDB 3.2 Manual — MongoDB Manual 3.2", *Docs.mongodb.com*, 2016. [Online]. Available: https://docs.mongodb.com/manual/. [Accessed: 09- Jun- 2016].

[24]"Node.js", *Nodejs.org*, 2016. [Online]. Available: https://nodejs.org/en/. [Accessed: 16- Mar- 2016].

[25]"Express - Node.js web application framework", *Expressjs.com*, 2016. [Online]. Available: http://expressjs.com/. [Accessed: 03- Mar- 2016].

[26]”Simple Sidebar -Bootstrap Sidebar Template -Start Bootstrap”, *Startbootstrap.com,* 2016. [Online]. Available: http://startbootstrap.com/template-overviews/simple-sidebar/ [Accessed: 03- Mar- 2016].

A. ANNEXES

A.1. Codi aplicació Login bàsic

lib/router.js

```
Router.configure({
       layoutTemplate:'layout'
});

Router.map(function () {
       this.route('home',{path:'/'});
       this.route('register', {path: '/register'});
});

Router.onBeforeAction(function(pause) {
  var routeName = this.route._path;
  if(routeName == '/register' && Meteor.userId()) {
    this.render('home');
    pause();
  } else{
    this.setLayout(this.next());
  }
});
```

controllers/userAccounts.js

```
Template.register.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    var username = template.find('#re-username').value;
    var password = template.find('#re-password').value;
    var password2 = template.find('#re-password2').value;
    if (password == password2){
      createAccount(username,password);
    }else{
      console.log("Error: Password incorrect");
      Router.go('register');
    }
  }
})

var createAccount = function(username,password){
  Accounts.createUser({
    username : username,
    password : password},
    function(err){
```

```
      if (err){
        console.log(err)
      }else{
        Router.go('home');
      }
    }
  )
}


Template.login.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    Meteor.loginWithPassword(
      template.find('#li-username').value,
      template.find('#li-password').value,
      function(err){
        if (err){
          console.log(err)
        }
      }
    )
  }
})

Template.logout.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    Meteor.logout(function(err){
      if (err){
        console.log(err)
      }else{
        Router.go('home');
      }
    })
  }
})
```

views/home.html

```
<template name='home'>
  {{> LogInLogOut}}
  {{#if currentUser}}
    <h1>Hola!</h1>
  {{/if}}
</template>

<template name='LogInLogOut'>
  {{#if loggingIn}}
    Logging in . . .
```

```
  {{else}}
       {{#if currentUser}}

    {{else}}
       {{> login}}
    {{/if}}
  {{/if}}
</template>

<template name='login'>
       <h1>Log In</h1>
       <form>
              <input type='text' id='li-username' placeholder='Username'/>
              <br>
              <input type='password' id='li-password' placeholder='Password'/>
              <br>
              <br>
              <button class="btn btn-success">Log In</button>
       </form>
       <br>
       <a class='link' href='/register'>Need an account? </a>
</template>

<template name='logout'>
       <form>
    {{ currentUser.username }}
              <button class="btn btn-danger">Log Out</button>
       </form>
</template>
```

views/layout.html

```
<template name='layout'>
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
  {{> navbar}}
  <div id='main' class="container text-center">
              {{> yield}}
       </div>
</template>

<template name='navbar'>
       <nav class="navbar navbar-default">
              <div class="navbar-header pull-left">
                     <a class="navbar-brand" href="/">Login</a>
              </div>

              <div class="navbar-header pull-right">
                     {{#if currentUser}}
                            {{> logout}}
```

```
                    {{/if}}
                </div>
        </nav>
</template>
```

views/register.html

```
<template name='register'>
  <h1>Register</h1>
        <form>
                <input type="text" id="re-username" placeholder='Username'/>
                <br>
                <input type="password" id="re-password" placeholder='Password'/>
                <br>
                <input type="password" id="re-password2" placeholder='Repeat Password'/>
                <br>
                <br>
    <button class="btn btn-success">Register</button> <a href="/" class="btn btn-
danger">Cancel</a>
        </form>
</template>
```

A.2. Codi del joc multijugador en temps real

lib/router.js

```
Router.configure({
        layoutTemplate:'layout'
});

Router.map(function () {
        this.route('home',{path:'/'});
        this.route('register', {path: '/register'});
        this.route('createRoom', {path: '/room/new'});
        this.route('searchRoom', {path: '/room/search'});
        this.route('playRoom', {path: 'room/:_id',
                data: function(){
                        return Rooms.findOne(this.params._id);
                }})
});

Router.onBeforeAction(function(pause){
  var routeName = this.route._path;
  if((routeName == '/register' && Meteor.userId()) || (! Meteor.userId() && (routeName !=
'/' && routeName != '/register'))) {
    this.render('home');
```

```
    pause();
  } else{
    this.setLayout(this.next());
  }
});
```

server/pack.js

```
Pack = {}

var preguntes = [
  {question: "2+2", correctAnswer: "4", otherAnswers: ["2","0","6"]},
  {question: "10+2", correctAnswer: "12", otherAnswers: ["10","5","20"]},
  {question: "100/25", correctAnswer: "4", otherAnswers: ["10","0","2"]},
  {question: "121/11", correctAnswer: "11", otherAnswers: ["21","2","5"]},
  {question: "-2 , 5 , -4 , 3 , -6 , ...", correctAnswer: "1", otherAnswers: ["0", "-3",
"-4"]},
  {question: "17 , 40 , 61 , 80 , 97 , ...", correctAnswer: "112", otherAnswers: ["100",
"114", "102"]},
  {question: "55 , 34 , 21 , 13 , 8 , ...", correctAnswer: "5", otherAnswers: ["4", "3",
"2", "1"]},
  {question: "7 , 21 , 14 , 42 , 28 , ...", correctAnswer: "84", otherAnswers: ["56",
"64", "76"]}
]

Pack.run=function(){
  Questions.remove();
  preguntes.forEach(function(question){
    Questions.insert(question);
  })
};
Meteor.startup(function(){
  if (Questions.find().count() != preguntes.length){
    Pack.run();
  }
});
```

models/gameCreation.js

```
GameFactory = {}

GameFactory.createGame = function(players){
  var questions = getQuestions();
  var playersScore = players.map(function(player){
    return {player : player, score : 0};
  });
  return {
    questions : questions,
    players : playersScore
  }
```

```
};

function getQuestions(){
  var questions = _.shuffle(Questions.find({}).fetch()).slice(0,3);
  var finalQuestions = questions.map(function(question){
    var responses = [{answer : question.correctAnswer, selected: false, correct : true}];
    var otherResponses = question.otherAnswers.forEach(function(answer){
      responses.push({answer: answer, selected: false, correct: false});
    });
    responses = _.shuffle(responses);
    return {question: question.question, responses: responses}
  })
  return finalQuestions;
}
```

models/games.js

```
Games = {}

Games.answerQuestion = function(roomCode, playerAnswer){
  var room = Rooms.findOne(roomCode);
  var game = room.game;
  var currentQuestion = game.questions[0];
  var currentPlayer = game.players.find(function(player){
    return player.player === Meteor.userId();
  });
  var selectedAnswer = currentQuestion.responses.find(function(answer){
    return answer.answer === playerAnswer;
  });
  selectedAnswer.selected = true;
  if (selectedAnswer.correct){
    currentPlayer.score +=50;
    game.questions.shift();
    if (game.questions.length ===0){
      room.playing = false;
      room.displayingResults = true;
    }
  }else{
    currentPlayer.score -=75;
  }
  return room;
}

Games.getResults = function(game){
  var players = game.players;
  return players.sort(function(a,b){
    return a.score < b.score
  });
}
```

models/questions.js

```
Questions = new Meteor.Collection('questions');
```

models/roomCreation.js

```
RoomFactory = {};

RoomFactory.createRoom = function(privacy){
  var room = {
    privacy : privacy,
    players : [],
    playersCount : 0,
    playing : false,
    displayingResults : false,
    full : false,
    game : null
  };
  RoomFactory.addPlayer(room, Meteor.userId());
  return room;
};

RoomFactory.addPlayer = function(room, playerId){
  if (!room.full){
    room.players.push(playerId);
    room.playersCount++;

    if (room.playersCount==4){
      room.full=true;
    }
    return true;
  }
  return false;
};

RoomFactory.removePlayer = function(room, playerId){
  room.players = _.without(room.players,playerId);
  room.playersCount = room.players.length;

  if (room.playersCount != 4){
    room.full = false;
  }
};

RoomFactory.startGame = function(room){
  var game = GameFactory.createGame(room.players);
  room.game = game;
  room.playing = true;
  room.displayingResults = false;
  return room;
```

```
};

RoomFactory.finishGame = function(room){
  room.game = null;
  room.playing = false;
  room.displayingResults = false;
  return room;
}
```

models/rooms.js

```
Rooms = new Meteor.Collection('rooms');

if (Meteor.isServer){
  Meteor.publish('rooms',function(){
    return Rooms.find({$or: [{full:false, playing: false},{players: this.userId}]});
  });

  Meteor.publish('users', function(){
    return Meteor.users.find();
  });

  Meteor.methods({
    createRoom : function(privacy){
      var room = RoomFactory.createRoom(privacy);
      return Rooms.insert(room, function(err,result){
        if (err){
          console.log(err);
        }
      });
    },
    leaveRoom : function(roomCode,playerId){
      var room = Rooms.findOne(roomCode);
      RoomFactory.removePlayer(room,playerId);
      Rooms.update(roomCode,room);
      if (room.playersCount==0){
        Rooms.remove(roomCode);
      }
    },
    getRandomRoom : function(){
      var room = Rooms.findOne({playing : false, full : false, privacy : false, players :
{$nin : [this.userId]}});
      if (!room){
        console.log("Error: cap room disponible");
      }else{
        return room;
      }
```

```
    },
    joinRoom : function(roomId,playerId){
      var room = Rooms.findOne(roomId);
      if (!room){
        console.log("Error: aquest roomId no existeix");
      }
      if (RoomFactory.addPlayer(room, playerId)){
        return Rooms.update(roomId, room);
      }else{
        console.log("Error: Room no disponible");
      }
    },
    startGame : function(roomId){
      var room = Rooms.findOne(roomId);
      updatedRoom = RoomFactory.startGame(room);
      Rooms.update(room._id, updatedRoom);
      return updatedRoom;
    },
    answerQuestion : function(roomCode, answer){
      var room = Games.answerQuestion(roomCode,answer);
      Rooms.update(room._id,room);
    },
    finishGame : function(roomCode){
      var room = Rooms.findOne(roomCode);
      updatedRoom = RoomFactory.finishGame(room);
      Rooms.update(room._id, updatedRoom);
    }
  })
};

if (Meteor.isClient) {
  Meteor.subscribe('rooms');
  Meteor.subscribe('users');
}
```

client/views/home.html

```
<template name='home'>
  {{> LogInLogOut}}
  {{#if currentUser}}
    {{> menu}}
  {{/if}}
</template>

<template name='LogInLogOut'>
  {{#if loggingIn}}
    Logging in . . .
  {{else}}
       {{#if currentUser}}
```

```
    {{else}}
       {{> login}}
    {{/if}}
  {{/if}}
</template>

<template name='login'>
       <h1>Log In</h1>
       <div class="form">
              <input type='text' class="form-control" id='li-username'
placeholder='Username'/>
              <br>
              <input type='password' class="form-control" id='li-password'
placeholder='Password'/>
              <br>
              <br>
              <button class="btn btn-success">Log In</button>
       </div>
       <br>
       <a class='link' href='/register'>Need an account? </a>
</template>

<template name='logout'>
       <form>
    {{ currentUser.username }}
              <button class="btn btn-danger">Log Out</button>
       </form>
</template>
```

client/views/layout.html

```
<template name='layout'>
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
  {{> navbar}}
  <div id='main' class="container text-center">
              {{> yield}}
       </div>
</template>

<template name='navbar'>
       <nav class="navbar navbar-default">
              <div class="navbar-header pull-left">
                     <a class="navbar-brand" href="/">Logic Game</a>
```

```
            </div>

            <div class="navbar-header pull-right">
                {{#if currentUser}}
                    {{> logout}}
                {{/if}}
            </div>
        </nav>
</template>
```

client/views/menu.html

```
<template name='menu'>
  {{#if publicRoom}}
            <a id='play-now' class="btn btn-primary">Play Now</a>
            <br>
            <br>
        {{/if}}
        <a href="/room/new" class="btn btn-primary">Create Game</a>
        <br>
        <br>
        <a href="/room/search" class="btn btn-primary">Search Game</a>
</template>
```

client/views/register.html

```
<template name='register'>
  <h1>Register</h1>
        <div class="form">
            <input type="text" class="form-control" id="re-username"
placeholder='Username'/>
            <br>
            <input type="password" class="form-control" id="re-password"
placeholder='Password'/>
    <br>
            <input type="password" class="form-control" id="re-password2"
placeholder='Repeat Password'/>
            <br>
    <button class="btn btn-success">Register</button> <a href="/" class="btn btn-
danger">Cancel</a>
        </div>
</template>
```

client/views/rooms.html

```
<template name='createRoom'>
  <h1>New Room</h1>
  <form>
    <input id='private_room' name='privacy' type='radio' value='private' />
    <label for='private_room'> Private </label>
    <br>
    <input id='public_room' name='privacy' type='radio' value='public' />
    <label for='public_room'> Public </label>
    <br>
    <a class='btn btn-danger' href='/'> Cancel </a>
    <button id='create-game' class='btn btn-success'> Create Game </button>
  </form>
</template>

<template name='searchRoom'>
  <h1> Join Room </h1>
  <form>
    <input type='text' id='room-code' placeholder='Room ID' />
    <br>
    <br>
    <a class='btn btn-danger' href='/'> Cancel </a>
    <button class='btn btn-primary' type='search' id='search'> Search </button>
  </form>
</template>

<template name="playRoom">
  {{#if playing}}
    {{> roomGame}}
  {{else}}
    {{#if displayingResults}}
      {{> roomResults}}
    {{else}}
      {{> waiting}}
    {{/if}}
  {{/if}}
</template>

<template name="waiting">
  {{#if full}}
    <h1>Ready to play</h1>
  {{else}}
    <h1>Waiting</h1>
  {{/if}}
  <p>Room Code:</p>
  <p>{{_id}}</p>
  <br>
  <p>Players {{playersCount}}/4</p>
    <ul style="list-style-type:none">
      {{#each players}}
```

```
        <li>
          {{userName this}}
        </li>
      {{/each}}
    </ul>

  <button class='btn btn-danger' id='leave-room'>Leave Room</button>
  {{#if canPlay playersCount}}
    <button class='btn btn-success' id='start-game'>Start Game</button>
  {{else}}
      <button class='btn btn-success' id='start-game' disabled>Start Game</button>
  {{/if}}

</template>

<template name="roomGame">
    <h1>{{currentQuestion.question}}</h1>
    <div class="row">
      {{#each currentQuestion.responses}}
        <div class="col-xs-3 col-md-6">
          {{#unless selected}}
            {{> questionAnswer}}
          {{/unless}}
        </div>
      {{/each}}
    </div>
    <h3>Score</h3>
    <div class="row">
    {{#each game.players}}
      <div class="col-xs-3">
        <h6>{{userName player}}</h6>
        <p>{{score}}</p>
      </div>
    {{/each}}
    </div>
</template>

<template name="questionAnswer">
  <button class="btn btn-primary" style="width:100%; margin-bottom:
1cm;">{{answer}}</button>
</template>

<template name="roomResults">
  <h1>Results</h1>
  {{#each sortResults}}
        <p> {{userName player}} : {{score}} </p>
  {{/each}}
  <button class="btn btn-danger" id='leave-room'>Leave Room</button>
  <button class='btn btn-success' id='finish'>Finish</button>
</template>
```

client/helpers/roomHelper.js

```
Handlebars.registerHelper('canPlay', function (count) {
  return (count > 1);
});

Template.roomGame.helpers({
  currentQuestion: function() {
    var room = this;
    var game = room.game;
    return game.questions[0];
  }
});

Template.roomResults.helpers({
  sortResults: function() {
    var room = this;
    return Games.getResults(room.game);
  }
})
```

client/helpers/username.js

```
Handlebars.registerHelper('userName', function (userId) {
  var user = Meteor.users.findOne(userId);
  return user.username;
});
```

client/controllers/menu.js

```
Template.menu.helpers({
  publicRoom : function(){
    return !!Rooms.findOne({
      playing : false,
      displayingResults : false,
      full : false,
      privacy : false
    });
  }
});

Template.menu.events({
  'click #play-now' : function(evt,template){
    evt.preventDefault();
    Meteor.call('getRandomRoom', function(err,result){
      if (result){
        Meteor.call('joinRoom', result._id,Meteor.userId(),function(error,res){
```

```
          if (err){
            console.log(error);
          }else{
            Router.go('playRoom',{_id : result._id});
          }
        });
      }else if (err){
        console.log(err);
      }
    });
  }
});
```

client/controllers/roomControl.js

```
Template.createRoom.events({
  'click #create-game' : function(evt,template){
    evt.preventDefault();
    var privacy= template.find('input[name=privacy]:checked').value == 'private';
    Meteor.call('createRoom',privacy,function(err,result){
      if (err){
        console.log(err);
      }else{
        Router.go('playRoom',{_id: result});
      }
    });
  }
});

Template.searchRoom.events({
  'click #search': function(evt,template){
    evt.preventDefault();
    var roomCode = template.find('#room-code').value;
    Meteor.call('joinRoom', roomCode,Meteor.userId(),function(err,result){
      if (err){
        console.log(err);
      }else{
        Router.go('playRoom',{_id: roomCode});
      }
    });
  }
});

Template.waiting.events({
  'click #leave-room' : function(evt, template) {
    evt.preventDefault();
    Meteor.call('leaveRoom',template.data._id,Meteor.userId());
    Router.go('home')
  },
```

```
  'click #start-game' : function(evt,template){
    evt.preventDefault();
    Meteor.call('startGame', template.data._id, function(err, result){
      if (err){
        console.log(err);
      }
    })
  }
});


Template.questionAnswer.events({
  'click button': function(evt,template){
    evt.preventDefault();
    var roomId = Template.parentData()._id;
    Meteor.call('answerQuestion', roomId, template.data.answer);
  }
})

Template.roomResults.events({
  'click #leave-room' : function(evt, template) {
    evt.preventDefault();
    Meteor.call('leaveRoom',template.data._id,Meteor.userId());
    Router.go('home')
  },
  'click #finish' : function(evt, template){
    evt.preventDefault();
    var roomId = Template.parentData()._id;
    Meteor.call('finishGame',roomId);
  }
});
```

client/controllers/usersAccounts.js

```
Template.register.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    var username = template.find('#re-username').value;
    var password = template.find('#re-password').value;
    var password2 = template.find('#re-password2').value;
    if (password == password2){
      createAccount(username,password);
    }else{
      console.log("Error: Password incorrect");
      Router.go('register');
    }
  }
})

var createAccount = function(username,password){
```

```
  Accounts.createUser({
    username : username,
    password : password},
    function(err){
      if (err){
        console.log(err)
      }else{
        Router.go('home');
      }
    }
  )
}

Template.login.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    Meteor.loginWithPassword(
      template.find('#li-username').value,
      template.find('#li-password').value,
      function(err){
        if (err){
          console.log(err)
        }
      }
    )
  }
})

Template.logout.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    Meteor.logout(function(err){
      if (err){
        console.log(err)
      }else{
        Router.go('home');
      }
    })
  }
})
```

A.3. Codi gestor de despeses per grups

lib/router.js

```
Router.configure({
	layoutTemplate:'layout'
});

Router.map(function () {
	this.route('home',{path:'/'});
	this.route('register', {path: '/register'});
	this.route('newExpense', {path: '/group/:_id/expense/new',
		data: function(){
			return Groups.findOne(this.params._id)
		}
	});
	this.route('group', {path: '/group/:_id',
		data: function(){
			return Groups.findOne(this.params._id);
		}
	});
	this.route('editGroup', {path: '/group/:_id/edit',
		data: function(){
			return Groups.findOne(this.params._id);
		}
	});
	this.route('newGroup', {path: '/group/:_id/new',
		data: function(){
			return Groups.findOne(this.params._id);
		}
	});
	this.route('userInfo', {path: '/user/:_id',
		data: function(){
			return PersonsTotal.findOne({userId : this.params._id});
		}
	});
	this.route('expenseInfo', {path: '/expense/:_id',
		data: function(){
			return Expenses.findOne(this.params._id);
		}
	});
});

Router.onBeforeAction(function(pause){
  var routeName = this.route._path;
  if((routeName == '/register' && Meteor.userId()) || (! Meteor.userId() && (routeName !=
'/' && routeName != '/register'))) {
    this.render('home');
	}
	this.next();
});
```

mobile-config.js

```
App.info({
  name: 'Money App',
  author: 'Oriol Vall',
  website: 'http://tfg-money-app.meteor.com/'
});

App.icons({
  // Android
  'android_xxhdpi': 'ic_logo_xxhdpi.png',
  'android_xxxhdpi': 'ic_logo_xxxhdpi.png',
  'android_mdpi': 'ic_logo_mdpi.png',
  'android_hdpi': 'ic_logo_hdpi.png',
  'android_xhdpi':'ic_logo_xhdpi.png'
});

App.setPreference('Orientation', 'portrait');
```

models/debts.js

```
Debts = new Mongo.Collection("debts");

if(Meteor.isServer){
  Meteor.publish("debts", function(){
    return Debts.find({userId : this.userId});
  });

  Meteor.methods({
    settleDebtUp : function(debt){
      Meteor.call("addExpense", "Settle Up",debt.money, this.userId, [debt.userId],
debt.group);
    },
    updateDebts : function(){
      Persons.find().forEach(
        function(user){
          var Array = [];
          Persons.find({$and : [{userId : user.userId},{money : {$lt : 0}}]}).forEach(
            function(debt){
              var person = Persons.findOne({$and: [{money: -debt.money}, {group:
debt.group}]});
              if (person){
                debt.money += person.money;
                Array.push(person);
              }else{
                Persons.find({$and: [{money: {$gt : 0}}, {group: debt.group}]},
{sort:{money:-1}}).forEach(
                  function(person){
```

```
                    if (debt.money!=0){
                      if (-debt.money>person.money){
                        debt.money += person.money;
                      }else{
                        person.money=-debt.money;
                        debt.money=0;
                      }
                      Array.push(person);
                    }
                  }
                );
              }
            }
          );
          Debts.update({userId : user.userId},{$set : {debts : Array}});
        }
      )
    }
  })
};

if(Meteor.isClient){
  Meteor.subscribe("debts");
}
```

models/expenses.js

```
Expenses = new Mongo.Collection("expenses");

if(Meteor.isServer){
  Meteor.publish("expenses", function(){
    return Expenses.find();
  })
  Meteor.publish('users', function(){
    return Meteor.users.find();
  });

  Meteor.methods({
    addExpense : function(desc, price, owner, debtors, groupId){
      price=parseFloat(price)
       Expenses.insert({
         description : desc,
         price : price,
         owner : owner,
         debtors : debtors,
         group : groupId
       });
       Meteor.call("updatePerson", owner, groupId, price);
       var splitPrice = (price / debtors.length);
       for (var i = 0; i < debtors.length; i++) {
```

```
        Meteor.call("updatePerson", debtors[i], groupId, -splitPrice);
      }
      Meteor.call("updateDebts");
    },
    deleteExpense : function(expenseId){
      var expense = Expenses.findOne(expenseId);
      if(expense){
        Expenses.remove(expenseId);
        Meteor.call("updatePerson", expense.owner, expense.group, -expense.price);
        var splitPrice = (expense.price / expense.debtors.length);
        for (var i = 0; i < expense.debtors.length; i++){
          Meteor.call("updatePerson", expense.debtors[i], expense.group, splitPrice);
        }
        Meteor.call("updateDebts");
      }
    }
  });
};

if(Meteor.isClient){
  Meteor.subscribe("expenses");
  Meteor.subscribe("users");
}
```

models/groups.js

```
Groups = new Mongo.Collection("groups");

if(Meteor.isServer){
  Meteor.publish("groups", function(){
    return Groups.find({members : this.userId});
  });

  Meteor.methods({
    newGroup:function(){
       Groups.insert({
         name : "New Group",
         members : [Meteor.userId()],
         oldMembers : [Meteor.userId()]
       }, function(err, groupId){
         Meteor.call("createPerson", Meteor.userId(), groupId, true);
       });
    },
    leaveGroup:function(groupId, userId){
      var group = Groups.findOne({_id:groupId});
      members = group.members;
      index = members.indexOf(userId);
      if (index > -1){
```

```
        members.splice(index,1);
      }
      Meteor.call("editGroup",groupId, group.name,members)
    },
    saveGroup:function(groupId, groupName, groupMembers){
      Groups.update(
        {_id : groupId},
        {$set: {name : groupName, members : groupMembers, oldMembers : groupMembers}},
function(err, result){
          for (var i = 0; i < groupMembers.length; i++) {
            Meteor.call("editPerson", groupMembers[i], groupId);
          }
        });
     },
     editGroup:function(groupId, groupName, groupMembers){
       Groups.update(
         {_id : groupId},
         {$set: {name : groupName, members : groupMembers}}, function(err, result){
           for (var i = 0; i < groupMembers.length; i++) {
             Meteor.call("editPerson", groupMembers[i], groupId);
           }
         });
      },
      cancelEdit:function(groupId){
        group=Groups.findOne({_id:groupId});
        if(group){
          for (var i=0; i < group.members.length; i++){
              if (group.oldMembers.indexOf(group.members[i])==-1){
                Meteor.call("removeMember", group._id, group.members[i]);
              }
          };
          for (var i=0; i<group.oldMembers.length;i++){
            if (group.members.indexOf(group.oldMembers[i])==-1){
              Meteor.call("addMember", group._id, group.oldMembers[i]);
            }
          }
          Groups.update(
            {_id : groupId},
            {$set: {members : group.oldMembers}}, function(err, result){
              for (var i = 0; i < group.oldMembers.length; i++) {
                Meteor.call("editPerson", group.oldMembers[i], groupId);
              }
          });
        }
     },
     addMember : function(groupId, userId){
       var group = Groups.findOne(groupId);
       members = group.members;
       members.push(userId);
       Meteor.call("editGroup",groupId, group.name, members)
     },
```

```
    removeMember : function(groupId, userId){
      var group = Groups.findOne(groupId);
      members = group.members;
      newMembers = [];
      for (var i = 0; i < members.length; i++) {
        if (members[i]!=userId){
          newMembers.push(members[i]);
        }else{
          Meteor.call("deletePerson", members[i], groupId);
        }
      };
      Meteor.call("editGroup",groupId, group.name, newMembers);
    },
    deleteAllExpenses:function(groupId){
      var cursor = Expenses.find({group : groupId});
      cursor.forEach(function(expense){
        Meteor.call("deleteExpense", expense._id);
      });
    },
    deleteGroup:function(groupId){
      Meteor.call("deleteAllExpenses", groupId);
      var group = Groups.findOne(groupId);
      for (var i = 0; i < group.members.length; i++) {
        Meteor.call("deletePerson", group.members[i], groupId);
      };
      Groups.remove(groupId);
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("groups");
}
```

models/persons.js

```
Persons = new Mongo.Collection("persons");
PersonsTotal = new Mongo.Collection("persons-total");

if(Meteor.isServer){
  Meteor.publish("persons", function(){
    return Persons.find();
  });
  Meteor.publish("persons-total", function(){
    return PersonsTotal.find({userId : this.userId});
  });
```

```
  Meteor.methods({
    updatePerson : function(id, group, money){
       Persons.update(
         {userId : id, group: group},
         {$inc : {money : money}
       });
       PersonsTotal.update(
         {userId : id},
         {$inc : {money : money}
       });
    },
    createPerson : function(id, groupId, admin){
      Persons.insert({
        userId : id,
        money : 0,
        group : groupId,
        admin : admin
      });
      var user = PersonsTotal.findOne({userId : id})
      if (user){
        PersonsTotal.update({userId : id}, {$push : {groups : groupId}});
      }else{
        PersonsTotal.insert({
          userId : id,
          money : 0,
          groups : [groupId]
        });
        Debts.insert({
          userId : id,
          debts : []
        })
      }
    },
    editPerson : function(userId, groupId){
      var member = Persons.findOne({userId : userId , group : groupId});
      if (member){
      } else{
        Meteor.call("createPerson", userId, groupId, false);
      }
    },
    deletePerson : function(id, groupId){
      Persons.remove({userId: id, group : groupId});
      PersonsTotal.update({userId : id}, {$pull: {groups : groupId}});
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("persons");
  Meteor.subscribe("persons-total");
}
```

client/views/expenses.html

```
<template name="newExpense">
  <h1 class="text-center">New Expense</h1>
       <div class="form">
      <div class="form-group">
        <label for="new-desc">Desciption</label>
        <input type="text" class="form-control" id="new-desc" placeholder='Description'/>
      </div>
      <div class="form-group">
        <label for="new-price">Price</label>
        <input type="number" class="form-control" id="new-price" min="0"
placeholder='Price'/>
      </div>
      <div class="form-group">
        <label for="new-owner">Who paid?</label>
        <select class="form-control" id="new-owner">
          {{#each members}}
            <option value={{this}}>{{userName this}}</option>
          {{/each}}
        </select>
      </div>
      <div class="form-group">
        <label>Share with...</label>
        <div id="checkboxes" class="checkbox">
          <label><input type="checkbox" id="all"><strong>Check all</strong></label>
          <br>
          <br>
          <div id="checkboxes2" class="checkbox">
            {{#each members}}
              <label><input type="checkbox" id="chk" value={{this}}> {{userName
this}}</label>
              <br>
            {{/each}}
          </div>
        </div>
      </div>
      <div class="text-center">
        <a href="/group/{{_id}}" class="btn btn-link">Cancel</a> <button class="btn btn-
success">Add</button>
      </div>
       </div>
</template>
```

client/views/groups.html

```
<template name="editGroup">
  <div class="text-center">
    <h1 style="display:inline;">{{name}}</h1><button class="btn btn-link"
id="cancel">Cancel</button>
    <br>
    <div class="form">
      <input type="text" class="form-control" id="new-name" placeholder='New Name'/>
      <br>
      <br>
      {{> chooseUsers}}
      <br>
      <button class="btn btn-danger" id="delete-group">Delete Group</button>
      <button class="btn btn-success" id="save-group">Save</button>
    </div>
  </div>
</template>

<template name="newGroup">
  <div class="text-center">
    <h1>{{name}}</h1>
    <div class="form">
      <input type="text" class="form-control" id="new-name" placeholder='New Name'/>
      <br>
      <br>
      {{> chooseUsers}}
      <br>
      <button class="btn btn-link" id="delete-group">Cancel</button>
      <button class="btn btn-success" id="save-group">Save</button>
    </div>
  </div>
</template>

<template name="group">
  <div class="text-center">
    <h1 style="display:inline">{{name}}</h1>
    {{#if admin}}
      <a href="/group/{{_id}}/edit" class="btn">Edit Group</a>
    {{else}}
      <button class="btn btn-danger" id="leave-group" disabled={{#if (notZero
currentUser._id _id)}}"disabled"{{/if}}>Leave Group</button>
      <br>
    {{/if}}
    <br>
    <a href="/group/{{_id}}/expense/new" class="btn btn-success">Add Expense</a>
    {{#each members}}
      {{> person}}
    {{/each}}
    <br>
    {{> allExpenses}}
  </div>
```

```
</template>

<template name="person">
  <div class="row">
    <div class="col-xs-6">
      <h3><strong>{{userName this}}</strong></h3>
    </div>
    <div class="col-xs-6">
      <h2 style="color:{{#if isPositive (Money this
../_id)}}green{{else}}red{{/if}}">{{splitNumber (Money this ../_id)}} €</h2>
    </div>
  </div>
</template>

<template name="allExpenses">
  <h2>Expenses</h2> <button class="btn btn-danger" id="delete-expenses">Delete
All</button>
  {{#each expenses (_id)}}
    {{> expenseRow}}
  {{/each}}
</template>

<template name="expenseRow">
  <div class="row">
    <div class="col-xs-6">
      <h3><strong><a href="/expense/{{_id}}">{{description}}</a></strong></h3>
    </div>
    <div class="col-xs-6">
      <h2>{{splitNumber price}} €</h2>
    </div>
  </div>
</template>
```

client/views/home.html

```
<template name='home'>
  <div class="text-center">
    {{> LogInLogOut}}
    {{#if currentUser}}
      {{> mainPage}}
    {{/if}}
  </div>
</template>

<template name='LogInLogOut'>
  {{#if loggingIn}}
    Logging in . . .
  {{else}}
       {{#if currentUser}}
```

```
    {{else}}
       {{> login}}
    {{/if}}
  {{/if}}
</template>

<template name='login'>
       <h1>Log In</h1>

               <input type='text' id='li-username' class="form-control"
placeholder='Username'/>
               <br>
               <input type='password' class="form-control" id='li-password'
placeholder='Password'/>
               <br>
               <br>
               <button class="btn btn-success">Log In</button>

       <br>
       <a class='link' href='/register'>Need an account? </a>
</template>

<template name='logout'>
       <form>
    <button class="btn btn-primary" id="user">{{currentUser.username}}</button>
               <button class="btn btn-danger" id="log-out">Log Out</button>
       </form>
</template>

<template name="mainPage">
  <button class="btn btn-success" id="new-group-link">New Group</button>
  <!--<a href="/group/new" class="btn btn-success"></a>-->
  {{#each groups}}
    <h2><strong><a href="/group/{{_id}}">{{name}}</a></strong></h2>
  {{/each}}
</template>
```

client/views/info.html

```
<template name="userInfo">
  <div class="text-center">
    <h1><strong>{{currentUser.username}}</strong></h1>
    <h2 style="color:{{#if isPositive money}}green{{else}}red{{/if}}">{{splitNumber
money}} €</h2>
    <hr>
    {{> debtsResum}}
    <hr>
    <div class="row">
      <div class="col-sm-6">
        <h3><strong>Expenses:</strong></h3>
        {{#each ownerExpenses}}
          <h3><a href="/expense/{{_id}}">{{description}}</a></h3>
        {{/each}}
      </div>
      <div class="col-sm-6">
        <h3><strong>Debts:</strong></h3>
        {{#each debtorExpenses}}
          <h3><a href="/expense/{{_id}}">{{description}}</a></h3>
        {{/each}}
      </div>
    </div>
  </div>
</template>

<template name="debtsResum">
  <h3><strong>Debts:</strong></h3>
    <div class="row">
      {{#with debtsMongo}}
        {{#each debts}}
          {{> debtInfo}}
        {{/each}}
      {{/with}}
    </div>
    <h3><strong>Profits:</strong></h3>
    {{#each profits}}
      {{> profitInfo}}
    {{/each}}
</template>

<template name="debtInfo">
  <div class="col-xs-6">
    <h4><span style="color:red">{{splitNumber money}} €</span> to <strong>{{userName
userId}}</strong> from <strong><a href="/group/{{group}}">{{groupName
group}}</a></strong></h4>
  </div>
  <div class="col-xs-6">
    <button class="btn btn-success">Settle Up</button>
  </div>
  <br>
```

```
  <br>
  <br>
</template>

<template name="profitInfo">
    <h4><span style="color:green">{{splitNumber money}} €</span> from <strong><a
href="/group/{{group}}">{{groupName group}}</a></strong></h4>
  <br>
  <br>
  <br>
</template>

<template name="expenseInfo">
  <div class="text-center">
    <h1><strong>{{description}}</strong></h1>
    <h2 style="color:green">{{splitNumber price}} €</h2>
    <div class="row">
      <div class="col-sm-6">
          <h3><strong>Who paid?</strong></h3>
          <h3>{{userName owner}}</h3>
      </div>
      <div class="col-sm-6">
        <h3><strong>Debtors:</strong></h3>
        {{#each debtors}}
          <h3>{{userName this}}</h3>
        {{/each}}
      </div>
    </div>
    <button id="back" class="btn btn-link">Cancel</button> <a href="/group/{{group}}"
class="btn btn-primary">View Group</a> <button class="btn btn-danger"
id="delete">Delete</button>
  </div>
</template>
```

client/views/layout.html

```
<template name='layout'>
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
  {{> navbar}}
  {{> sAlert}}
  <div id='main' class="container">
              {{> yield}}
       </div>
</template>

<template name='navbar'>
       <nav class="navbar navbar-default">
              <div class="navbar-header pull-left">
      <div class="row">
        <div class="col-xs-2">
```

```
          <a href="/"><img src="/images/logo.png" style="width:100%"/></a>
        </div>
        <div class="col-xs-9">
          <a class="navbar-brand" href="/">Money App</a>
        </div>
      </div>
              </div>

              <div class="navbar-header pull-right">
                     {{#if currentUser}}
                            {{> logout}}
                     {{/if}}
              </div>
       </nav>
</template>
```

client/views/main.html

```
<head>
  <title>Money App</title>
  <link rel="icon" type="image/png" href="/images/logo.png"/>
</head>
```

client/views/register.html

```
<template name='register'>
  <div class="text-center">
    <h1>Register</h1>
       <div class="form">
              <input type="text" class="form-control" id="re-username"
placeholder='Username'/>
              <br>
              <input type="password" class="form-control" id="re-password"
placeholder='Password'/>
              <br>
              <input type="password" id="re-password2" class="form-control"
placeholder='Repeat Password'/>
              <br>
              <br>
      <a href="/" class="btn btn-link">Cancel</a> <button class="btn btn-
success">Register</button>
       </div>
  </div>
</template>
```

client/views/search.html

```
<template name="chooseUsers">
    <div class="input-group">
      <input type="text" class="form-control" id="user-name-text" placeholder='Search
username'/>
      <span class="input-group-btn">
        <button class="btn btn-default" type="button">
          <span class="glyphicon glyphicon-search" id="search-user"></span>
        </button>
      </span>
    </div>
  <br>
  <h4>Members:</h4>
  <div class="row">
    {{#each members}}
      {{> memberRow}}
    {{/each}}
  </div>
</template>

<template name="memberRow">
  <div class="col-xs-6">
    {{userName this}}
  </div>
  <div class="col-xs-6">
    {{#if equal currentUser._id this}}
      <button class="btn btn-danger" id="delete-member" disabled>X</button>
    {{else}}
      <button class="btn btn-danger" id="delete-member" disabled={{#if (notZero this
../_id)}}"disabled"{{/if}}>X</button>
    {{/if}}
  </div>
  <br>
</template>
```

client/helpers/helper.js

```
Template.group.helpers({
  admin: function(){
    return Persons.findOne({userId: Meteor.userId(), group: this._id, admin:true});
  }
});

Template.editGroup.helpers({
  users: function(){
    return Meteor.users.find();
  }
});
```

```
Template.debtsResum.helpers({
  profits : function(){
    return Persons.find({userId:Meteor.userId(), money:{$gt:0}});
  }
});

Template.mainPage.helpers({
  groups : function(){
    return Groups.find();
  }
});

Template.allExpenses.helpers({
  expenses : function(groupId){
    return Expenses.find({group : groupId})
  }
});

Template.userInfo.helpers({
  ownerExpenses: function(){
    return Expenses.find({owner: this.userId})
  },
  debtorExpenses: function(){
    return Expenses.find({debtors: this.userId})
  }
});

Template.debtsResum.helpers({
  debtsMongo : function(){
    return Debts.findOne();
  }
});
```

client/helpers/registerHelpers.js

```
Template.registerHelper("userName", function(userId){
  var user = Meteor.users.findOne(userId);
  if(user){
    return user.username;
  }
});

Template.registerHelper("equal", function(a, b){
  return (a==b);
});

Template.registerHelper("groupName", function(groupId){
  var group = Groups.findOne(groupId);
```

```
  if(group){
    return group.name;
  }
});

Template.registerHelper("isPositive", function(number){
  return (number >= 0);
});

Template.registerHelper("isMember", function(userId, groupId){
    var user = Persons.findOne({userId : userId, group : groupId});
    return user;
});

Template.registerHelper("notZero", function(userId, groupId){
  if (userId && groupId){
    var user = Persons.findOne({userId : userId, group : groupId});
    if (user){
      return (user.money != 0);
    }else{
      return false;
    }
  }
});

Template.registerHelper("splitNumber", function(number){
  if(number){
    return number.toFixed(2);
  }else{
    return number;
  }
});

Template.registerHelper("Money", function(userId, groupId){
  if (userId && groupId){
    var user = Persons.findOne({userId : userId, group : groupId});
    if (user){
      return user.money;
    }
  }
});
```

client/helpers/sAlert.js

```
Meteor.startup(function () {
    sAlert.config({
        effect: 'slide',
        position: 'top-right',
        timeout: 3000,
        html: false,
        onRouteClose: true,
        stack: true,
        offset: 0,
        beep: false,
        onClose: _.noop
    });
});
```

client/controllers/expensesController.js

```
Template.newExpense.events({
  "click #all": function(event, template){
    state=event.toElement.checked;
    $("#checkboxes input").each(function(){
      this.checked=state;
    });
  },
  "click button": function(event, template){
     event.preventDefault();
     var desc = template.find("#new-desc").value;
     var price = template.find("#new-price").value;
     var owner = template.find("#new-owner").value;
     var group = template.data._id;
     var debtors = [];
     $("#checkboxes2 input:checked").each(function(){
       debtors.push($(this).attr('value'))
     });
     Meteor.call('addExpense',desc, price, owner, debtors, group);
     Router.go('/group/'+group);
  }
});
```

client/controllers/groupsController.js

```
Template.group.events({
  "click #leave-group": function(event, template){
    event.preventDefault();
    Meteor.call("leaveGroup", template.data._id, Meteor.userId());
    Router.go('/');
  }
});
```

```
Template.editGroup.events({
  "click #delete-group": function(event, template){
     event.preventDefault();
     swal({
       title: "Are you sure?",
       text: "You will not be able to recover this group!",
       type: "warning",
       showCancelButton: true,
       confirmButtonColor: "#DD6B55",
       confirmButtonText: "Yes, delete it!",
       closeOnConfirm: false
     }, function(){
       Meteor.call("deleteGroup", template.data._id, function(error, result){
         if(error){
           swal("Error!",error,"error");
         }else{
          swal("Group deleted!","","success");
         };
       });
      Router.go('/');
     });
   },
  "click #save-group, keypress input": function(evt, template){
    if ((evt.type === 'click') || (evt.type === 'keypress' && evt.which === 13 &&
evt.target.id == "new-name") ) {
      var name = template.find("#new-name").value;
      if (name==""){
        name=template.data.name;
      };
      Meteor.call('saveGroup',template.data._id, name, template.data.members);
      Router.go('/group/'+template.data._id);
    }
  },
  "click #cancel":function(event,template){
    Meteor.call("cancelEdit", template.data._id);
    history.back();
  }
});

Template.newGroup.events({
  "click #delete-group": function(event, template){
     event.preventDefault()
       Meteor.call("deleteGroup", template.data._id, function(error, result){
         if(error){
           swal("Error!",error,"error");
         }else{
           Router.go('/');
         }
       });
   },
```

```
  "click #save-group, keypress input": function(evt, template){
    if ((evt.type === 'click') || (evt.type === 'keypress' && evt.which === 13 &&
evt.target.id == "new-name") ) {
      var name = template.find("#new-name").value;
      if (name==""){
        name=template.data.name;
      }
      Meteor.call('saveGroup',template.data._id, name, template.data.members);
      Router.go('/group/'+template.data._id);
    }
  }
});

Template.allExpenses.events({
  "click #delete-expenses": function(event, template){
    swal({
      title: "Are you sure?",
      text: "You will not be able to recover the expenses!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
    }, function(){
      Meteor.call("deleteAllExpenses", template.data._id, function(error, result){
        if(error){
          swal("Error!",error,"error");
        }else{
         swal("Expenses deleted!","","success");
        };
      });
    });
  }
});
```

client/controller/infoController.js

```
Template.expenseInfo.events({
  "click #delete": function(event, template){
    event.preventDefault();
    swal({
      title: "Are you sure?",
      text: "You will not be able to recover this expense!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
    }, function(){
```

```
      Meteor.call("deleteExpense", template.data._id, function(error, result){
        if(error){
          swal("Error!",error,"error");
        }else{
          swal("Expense deleted!","","success");
        };
      });
      history.back();
    });
  },
  "click #back":function(event,template){
    history.back();
  }
});

Template.debtInfo.events({
  "click button": function(event, template){
    Meteor.call("settleDebtUp", template.data);
  }
});
```

client/controllers/mainController.js

```
Template.mainPage.events({
  "click #new-group-link": function(event, template){
    Meteor.call("newGroup", function(err, result){
      if (err){
        swal("Error!", err, "error");
      }else{
        var group=Groups.findOne({name: "New Group", members: [Meteor.userId()]});
        Router.go('/group/'+group._id+'/new');
      }
    });
  }
});
```

client/controllers/searchController.js

```
Template.chooseUsers.events({
  'click #search-user, keypress input': function(e,template){
    if ((e.type === 'click') || (e.type === 'keypress' && e.which === 13) ) {
      var userName = template.find('#user-name-text').value;
      var user = Meteor.users.findOne({username:userName});
      if (user){
        if (Persons.findOne({userId : user._id, group : template.data._id})){
          swal("Error!", "User already in this group", "error");
        }else{
          Meteor.call("addMember", template.data._id, user._id);
```

```
          e.target.value="";
        }
      }else{
        swal("Error!", "User not found", "error");
      }
    }
  }
});

Template.memberRow.events({
  "click #delete-member": function(event, template){
    Meteor.call("removeMember", Template.parentData(1)._id, template.data);
  }
});
```

client/controllers/usersAccounts.js

```
Template.register.events({
  'click button, keypress input' : function(evt,template){
    if ((evt.type === 'click') || (evt.type === 'keypress' && evt.which === 13) ) {
      var username = template.find('#re-username').value;
      var password = template.find('#re-password').value;
      var password2 = template.find('#re-password2').value;
      if (password == password2){
        createAccount(username,password);
      }else{
        swal("Warning", "Incorrect password", "warning");
        Router.go('register');
      }
    }
  }
});

var createAccount = function(username,password){
  Accounts.createUser({
    username : username,
    password : password},
    function(err){
      if (err){
        swal("Error!", err.reason, "error");
      }else{
        Meteor.call("createPerson", Meteor.userId());
        Router.go('home');
      }
    }
  )
}

Template.login.events({
  'click button, keypress input' : function(evt,template){
```

```
    if ((evt.type === 'click') || (evt.type === 'keypress' && evt.which === 13) ) {
      Meteor.loginWithPassword(
        template.find('#li-username').value,
        template.find('#li-password').value,
        function(err){
          if (err){
            swal("Error!", err.reason, "error");
          }
        }
      )
    }
  }
})

Template.logout.events({
  'click #log-out' : function(evt,template){
    evt.preventDefault();
    Meteor.logout(function(err){
      if (err){
        swal("Error!", err.reason, "error");
      }else{
        Router.go('home');
      }
    })
  },
  'click #user' : function(evt,template){
    evt.preventDefault();
    Router.go('/user/'+Meteor.userId())
  }
})
```

A.4. Codi app per organitzar colles castelleres

lib/router.js

```
Router.configure({
       layoutTemplate:'capdecollaLayout'
});

Router.map(function () {
       this.route('main',{path:'/capde'});
       this.route('pinya',{path:'/capde/pinya'});
       this.route('preguntes',{path:'/capde/esdeveniments'});
       this.route('preguntaInfo',{path:'/capde/esdeveniments/:_id',
              data:function(){
                     return Preguntes.findOne({_id:this.params._id});
              }
```

```
        });
        this.route('listCastellers',{path:'/capde/castellers'});
        this.route('capdeEdit',{path:'/capde/casteller/:_id',
                data:function(){
                        return Meteor.users.findOne({_id:this.params._id});

                }
        });
});

Router.route('/', function() {
        this.layout('layout');
        this.render('home')
});

Router.route('/register', function() {
        this.layout('layout');
        this.render('register')
});

Router.route('/edit', function() {
        this.layout('layout');
        this.render('edit')
});

if(Meteor.isClient){
        Session.setDefault("membres", []);
}
```

server/startup.js

```
var colles = [
  {nom:'Tirallongues', psw:"1234"},
  {nom:'Penjats', psw:"1234"}
]

Meteor.startup(function(){

  if (Colles.find().count() != colles.length){
    Colles.remove();
    colles.forEach(function(colla){
      Colles.insert(colla);
      if(Meteor.users.findOne({username:"admin"+colla.nom})){

      }else{
        Accounts.createUser({
          username: "admin"+colla.nom,
          password: colla.psw,
          profile:{
```

```
            nom:"",
            colla: colla.nom,
            admin:true
          }
        })
      }
    })
  }
});
```

models/colles.js

```
Colles = new Mongo.Collection("colles");

if(Meteor.isServer){
  Meteor.publish("colles", function(){
    return Colles.find();
  });
}

if(Meteor.isClient){
  Meteor.subscribe("colles");
}
```

models/preguntes.js

```
Preguntes = new Mongo.Collection("preguntes");

if(Meteor.isServer){
  Meteor.publish("preguntes", function(){
    var user=Meteor.users.findOne({"_id":this.userId});
    if (user){
      colla = user.profile.colla;

      return Preguntes.find({colla: colla});
    }
  });

  Meteor.methods({
    addCasteller:function(userId, colla){
      Preguntes.update({colla:colla},{$push: {nsnc:userId}},{multi:true});
    },
    addPregunta:function(pregunta, membres){
      var user=Meteor.users.findOne({"_id":this.userId});
      if (user){
        colla = user.profile.colla;
```

```
        Preguntes.insert({
          pregunta : pregunta,
          colla : colla,
          si : [],
          no : [],
          nsnc : membres
        })
      }
    },
    deletePregunta:function(preguntaId){
      Preguntes.remove({_id:preguntaId});
    },
    updatePregunta:function(preguntaId, value){
      Preguntes.update({_id:preguntaId}, {$pull:{
        si : this.userId,
        no : this.userId,
        nsnc:this.userId
      }}, function(error, result){
        if(error){
          console.log(error);
        }
      });
      if (value=="true"){
        Preguntes.update({_id:preguntaId}, {$push:{si:this.userId}});
      }else{
        console.log("hola");
        Preguntes.update({_id:preguntaId}, {$push:{no:this.userId}});
      }
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("preguntes");
}
```

models/users.js

```
Push.allow({
  send: function(userId, notification) {
    return true;
  }
});

if(Meteor.isServer){
  Meteor.users.allow({
    update: function(userId, param, fields, modifier){
      return Meteor.users.findOne({_id:userId, "profile.admin":true});
    }
  });
```

```
  Meteor.publish('users', function(){
    return Meteor.users.find();
  });
}

if(Meteor.isClient){
  Meteor.subscribe("users");
}
```

client/views/capdecolla.html

```
<template name="capdecollaLayout">

  {{#if loggingIn}}
    Logging in . . .
  {{else}}
       {{#if currentUser}}
      {{#if currentUser.profile.admin}}
        <div id="wrapper">
          {{> navBar}}
          <div id="page-content-wrapper">
              <button id="back" class="btn btn-danger">Sortir</button>
              <br>
              <br>
              {{#if itemsReady}}
                {{> yield}}
              {{else}}
                Carregant ...
              {{/if}}
          </div>
        </div>
      {{else}}
        <div class="container">
            {{> casteller}}
        </div>
      {{/if}}
    {{else}}
      <div class="container">
       {{> capdeLogin}}
      </div>
    {{/if}}
  {{/if}}
</template>

<template name="navBar">
  <div id="sidebar-wrapper">
    <ul class="sidebar-nav">
```

```
      <li class="sidebar-brand">
        <a href="/">Auto Pinyes</a>
      </li>
      <li>
        <a href="/capde/castellers">Castellers</a>
      </li>
      <li>
        <a href="/capde/esdeveniments">Esdeveniments</a>
      </li>
      <li>
        <a href="/capde/pinya">Pinya</a>
      </li>
    </ul>
  </div>
</template>

<template name="main">
      <h1>Auto Pinyes <strong>{{currentUser.profile.colla}}</strong></h1>
</template>

<template name="buttonGroup">
  <div class="btn-group" data-toggle="buttons">
    <label class="btn btn-primary active">
      <input type="radio" name="options" value="1" autocomplete="off" checked> Pilar
    </label>
    <label class="btn btn-primary">
      <input type="radio" name="options" value="2" autocomplete="off"> Torre
    </label>
    <label class="btn btn-primary">
      <input type="radio" name="options" value="3" autocomplete="off"> Castell de 3
    </label>
    <label class="btn btn-primary">
      <input type="radio" name="options" value="4" autocomplete="off"> Castell de 4
    </label>
    <label class="btn btn-primary">
      <input type="radio" name="options" value="5" autocomplete="off"> Castell de 5
    </label>
  </div>
</template>

<template name="buttonGroup2">
  <div class="btn-group" data-toggle="buttons">
    <label class="btn btn-warning active">
      <input type="radio" name="options" value="1" autocomplete="off" checked> Baixos
    </label>
    <label class="btn btn-warning">
      <input type="radio" name="options" value="2" autocomplete="off"> Agulles
    </label>
    <label class="btn btn-warning">
      <input type="radio" name="options" value="3" autocomplete="off"> Crosses
    </label>
```

```
    <label class="btn btn-warning">
      <input type="radio" name="options" value="4" autocomplete="off"> Contraforts
    </label>
    <label class="btn btn-warning">
      <input type="radio" name="options" value="5" autocomplete="off"> Mans
    </label>
    <label class="btn btn-warning">
      <input type="radio" name="options" value="6" autocomplete="off"> Vents
    </label>
    <label class="btn btn-warning">
      <input type="radio" name="options" value="7" autocomplete="off"> Laterals
    </label>
    <label class="btn btn-warning">
      <input type="radio" name="options" value="8" autocomplete="off"> Altres
    </label>
  </div>
</template>

<template name="capdeLogin">
  <a href="/" class="btn btn-link">Cancelar</a>
  <h1>Cap de Colla</h1>
  <div class="form-group">
    <label for="nom">Colla</label>
    <br>
    <select class="form-control" id="colla">
      {{#each colles}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="form-group">
    <label for="nom">Contrasenya</label>
    <input type="password" class="form-control" id="password" placeholder="Contrasenya">
  </div>
  <button class="btn btn-success">OK</button>
</template>

<template name="listCastellers">
  <h1>Castellers</h1>
  <button id="add" class="btn btn-success">Nou casteller</button>
  <br>
  <br>
  {{#each castellers}}
    <span style="font-size:150%"><a
href="/capde/casteller/{{_id}}">{{profile.nom}}</a></span>
    <br>
  {{/each}}
</template>
```

client/views/castellers.html

```
<template name="home">
  {{#if loggingIn}}
    Logging in . . .
  {{else}}
       {{#if currentUser}}
      {{#if currentUser.profile.admin}}
        <script charset="utf-8">
          Router.go('/capde');
        </script>
      {{else}}
        {{#if itemsReady}}
          {{> casteller}}
        {{/if}}
      {{/if}}
    {{else}}
       {{> login}}
    {{/if}}
  {{/if}}
</template>

<template name="casteller">
  <button class="btn btn-danger" id="back">sortir</button>
  <h1><strong>{{currentUser.profile.nom}}</strong><span class="btn btn-link"><a
href="/edit">Editar</a></span></h1>
  <hr>
  <div class="row">
    <div class="col-xs-6">
      <strong>Pregunta</strong>
    </div>
    <div class="col-xs-3">
      <strong>SI</strong>
    </div>
    <div class="col-xs-3">
      <strong>NO</strong>
    </div>
  </div>
  {{#each preguntes}}
    {{> preguntaAnswer}}
  {{/each}}
  <br><br>
  <button class="btn btn-success" id="save">Guardar</button>
</template>

<template name="login">
  <h1>Assistència</h1>
  <div class="form-group">
    <label for="nom">Nom</label>
    <input type="text" class="form-control" id="nom" placeholder="Nom">
  </div>
  <div class="form-group">
```

```
    <label for="nom">Colla</label>
    <br>
    <select class="form-control" id="colla">
      {{#each colles}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="text-center">
    <button id="sign-in" class="btn btn-success text-center">OK</button>
    <div class="row">
      <div class="col-xs-6">
        <a href="/register">Registrar-se</a>
      </div>
      <div class="col-xs-6">
        <a href="/capde">Cap de Colla</a>
      </div>
    </div>
  </div>
</template>

<template name="edit">
  <a href="/" class="btn btn-link">Cancelar</a>
  <h1>Editar</h1>
  <div class="form-group">
    <label for="nom">Nom</label>
    <input type="text" class="form-control" id="nom" value="{{currentUser.profile.nom}}"
disabled>
  </div>
  <div class="form-group">
    <label for="alçada">Alçada (cm)</label>
    <input type="number" class="form-control" id="alçada" placeholder="Alçada"
value="{{currentUser.profile.alçada}}">
  </div>

  <div class="form-group">
    <label for="pos1">Posició 1:</label>
    <select class="form-control" id="pos1">
      <option
value={{currentUser.profile.posicio1}}>{{currentUser.profile.posicio1}}</option>
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="form-group">
    <label for="pos2">Posició 2:</label>
    <select class="form-control" id="pos2">
      <option
value={{currentUser.profile.posicio2}}>{{currentUser.profile.posicio2}}</option>
      {{#each posicions}}
```

```
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="form-group">
    <label for="pos3">Posició 3:</label>
    <select class="form-control" id="pos3">
      <option
value={{currentUser.profile.posicio3}}>{{currentUser.profile.posicio3}}</option>
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>

  <button class="btn btn-success">Guardar</button>
</template>

<template name="capdeEdit">
  <h1>{{profile.nom}}</h1>
  <div class="form-group">
    <label for="alçada">Alçada (cm)</label>
    <input type="number" class="form-control" id="alçada" placeholder="Alçada"
value="{{profile.alçada}}">
  </div>


  <div class="form-group">
    <label for="pos1">Posició 1:</label>
    <select class="form-control" id="pos1">
      <option value={{profile.posicio1}}>{{profile.posicio1}}</option>
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="form-group">
    <label for="pos2">Posició 2:</label>
    <select class="form-control" id="pos2">
      <option value={{profile.posicio2}}>{{profile.posicio2}}</option>
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="form-group">
    <label for="pos3">Posició 3:</label>
    <select class="form-control" id="pos3">
      <option value={{profile.posicio3}}>{{profile.posicio3}}</option>
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
```

```
      {{/each}}
    </select>
  </div>


  <button class="btn btn-success">Guardar</button>
</template>
```

client/views/chart.html

```
<template name="chart">
  <div style="width:100%; white-space:nowrap;">
    <span style="display: inline-block; vertical-align: top; padding: 5px; width:100px">
      <div id="myGuests" style="border: solid 1px black; height: 500px"></div>
    </span>
    <span style="display: inline-block; vertical-align: top; padding: 5px; width:80%">
      <div id="myDiagramDiv" style="border: solid 1px black; height: 500px"></div>
    </span>
  </div>




  <script charset="utf-8">
    $ = go.GraphObject.make;
    myDiagram =
    $(go.Diagram, "myDiagramDiv",
    {
      allowDragOut: true,
      allowDrop: true,
      allowClipboard: false,
      allowRotate:false,
      allowMove:false,
      initialContentAlignment: go.Spot.Center,
      "undoManager.isEnabled": true
    });

    //template for people
    myDiagram.nodeTemplateMap.add("",
      $(go.Node, "Auto",
      { background: "transparent" },
      new go.Binding(
        "layerName",
        "isSelected",
        function(s) {
          return s ? "Foreground" : "";
        }).ofObject(),
        { locationSpot: go.Spot.Center },
```

```
        new go.Binding("location", "loc", go.Point.parse).makeTwoWay(go.Point.stringify),
        new go.Binding("text", "key"),
        {
          mouseDragEnter: function(e, node, prev) { highlightSeats(node, true); },
          mouseDragLeave: function(e, node, next) { highlightSeats(node, false); },
          mouseDrop: function(e, node) { assignPeopleToSeats(node,
node.diagram.selection, e.documentPoint); }
        },
        $(go.Shape, "Rectangle",{ fill: "blanchedalmond", stroke: null },new
go.Binding("fill", "color")),
        $(go.Panel, "Viewbox",
          { desiredSize: new go.Size(40, 40) },
          $(go.TextBlock, { margin: 2, desiredSize: new go.Size(60, NaN), textAlign:
"center", stroke: "darkblue" },
          new go.Binding("text", "", function(data) {
            var color = "blanchedalmond";
            var s = data.key;
            if (data.plus) s += " +" + data.plus.toString();
              return s;
            })
          )
        )
      )
    );

    function color(data) {
      //return "blanchedalmond"
      new go.Binding("text", "", function(data) {
        var s = "blanchedalmond";
        if (data.color) s = data.color;
        return s;
      })
    }


    //seat element
    function Seat(number,name, align, focus) {
      if (name=="B"){
        color="#FF2828"
      }else if(name=="A"){
        color="#FEFF8E"
      }else if(name=="C"){
        color="#FB856B"
      }else if(name=="CF"){
        color="#CAFF55"
      }else if(name=="M"){
        color="#A18EFF"
      }else if(name=="V"){
        color = "#8EFFBA"
      }else if(name=="L"){
        color="#FF8EFF"
```

```
      }else{
        color = "burlywood"
      }
      if (typeof align === 'string') align = go.Spot.parse(align);
      if (!align || !align.isSpot()) align = go.Spot.Right;
      if (typeof focus === 'string') focus = go.Spot.parse(focus);
      if (!focus || !focus.isSpot()) focus = align.opposite();
      return $(go.Panel, "Spot",
        { name: number.toString(), alignment: align, alignmentFocus: focus, data:color},
        $(go.Shape, "Circle",
        { name: "SEATSHAPE", desiredSize: new go.Size(40, 40), fill: color, stroke:
"white", strokeWidth: 2 },
        new go.Binding("fill")),
        $(go.TextBlock, name,
        new go.Binding("angle", "angle", function(n) { return -n; }))
      );
    }

    //table style
    function tableStyle() {
      return [
        { background: "transparent" },
        { layerName: "Background" },
        { locationSpot: go.Spot.Center, locationObjectName: "TABLESHAPE" },
        new go.Binding("location", "loc", go.Point.parse).makeTwoWay(go.Point.stringify),
        { rotatable: true },
        new go.Binding("angle").makeTwoWay(),
        {
          mouseDragEnter: function(e, node, prev) { highlightSeats(node, true); },
          mouseDragLeave: function(e, node, next) { highlightSeats(node, false); },
          mouseDrop: function(e, node) { assignPeopleToSeats(node,
node.diagram.selection, e.documentPoint); }
        }
      ];
    }

    //table with 8 seats
    myDiagram.nodeTemplateMap.add("TableR8",
      $(go.Node, "Spot", tableStyle(),
        $(go.Panel, "Spot",
          $(go.Shape, "Circle",
            { name: "TABLESHAPE", desiredSize: new go.Size(20, 20), fill: "transparent",
stroke: null },
            new go.Binding("desiredSize", "size",
go.Size.parse).makeTwoWay(go.Size.stringify),
            new go.Binding("fill")
          ),
          $(go.TextBlock, { editable: true, font: "bold 11pt sans-serif" },
          new go.Binding("text", "name").makeTwoWay(),
          new go.Binding("angle", "angle", function(n) { return -n; }))
        ),
```

```
      Seat(1,"B", "0 0", "0 0"),

      Seat(2,"A", "0 0", "0 1"),
      Seat(3,"M", "0 0", "0 2"),
      Seat(4,"M", "0 0", "0 3"),
      Seat(5,"M", "0 0", "0 4"),

      Seat(6,"CF", "0 2", "0 0"),
      Seat(7,"M", "0 4", "0 0"),
      Seat(8,"M", "0 6", "0 0"),
      Seat(9,"M", "0 8", "0 0"),

      Seat(10,"C", "2 0", "0 0"),
      Seat(11,"V", "4 0", "0 0"),
      Seat(12,"V", "6 0", "0 0"),
      Seat(13,"V", "8 0", "0 0"),

      Seat(14,"C", "0 0", "1 0"),
      Seat(15,"V", "0 0", "2 0"),
      Seat(16,"V", "0 0", "3 0"),
      Seat(17,"V", "0 0", "4 0"),

      Seat(18,"L", "2 2", "0 0"),
      Seat(19,"L", "3.5 3.5", "0 0"),
      Seat(20,"L", "5 5", "0 0"),

      Seat(21,"L", "0 0", "1 1"),
      Seat(22,"L", "0 0", "1.75 1.75"),
      Seat(23,"L", "0 0", "2.5 2.5"),

      Seat(24,"L", "2 0", "0 1"),
      Seat(25,"L", "3.5 0", "0 1.75"),
      Seat(26,"L", "5 0", "0 2.5"),

      Seat(27,"L", "0 2", "1 0"),
      Seat(28,"L", "0 3.5", "1.75 0"),
      Seat(29,"L", "0 5", "2.5 0")
    )
  );


  myDiagram.mouseDrop = function(e) {
    e.diagram.selection.each(function(n) {
      if (isPerson(n)) unassignSeat(n.data);
    });
  };

  myDiagram.addDiagramListener("ExternalObjectsDropped", function(e) {
    if (!e.subject.any(isTable)) {
```

```
        myGuests.commandHandler.deleteSelection();
      }
    });

    myDiagram.addDiagramListener("SelectionDeleted", function(e) {
      // no-op if deleted by myGuests' ExternalObjectsDropped listener
      if (myDiagram.disableSelectionDeleted) return;
      // e.subject is the myDiagram.selection collection
      e.subject.each(function(n) {
          if (isPerson(n)) {
            myGuests.model.addNodeData(myGuests.model.copyNodeData(n.data));
          }
        });
    });

    //create tables
    myDiagram.model = new go.GraphLinksModel([
      {"key":1, "category":"TableR8", "name":"", "guests":{}, "loc":"143.5 58"}
    ]);

    //guests
    myGuests =
      $(go.Diagram, "myGuests",
        {
          layout: $(go.GridLayout,
                    {
                      wrappingColumn: 1,  // limit items to one column
                      sorting: go.GridLayout.Ascending  // sort by Node.text value
                    }),
          allowDragOut: true,  // to myDiagram
          allowDrop: true  // from myDiagram
        });

    myGuests.nodeTemplateMap = myDiagram.nodeTemplateMap;

    myGuests.model = new go.GraphLinksModel(UI._globalHelpers['membres']());

    myGuests.model.undoManager = myDiagram.model.undoManager  // shared UndoManager!

    // To simulate a "move" from the Diagram back to the Palette, the source Node must be
deleted.
    myGuests.addDiagramListener("ExternalObjectsDropped", function(e) {
      // e.subject is the myGuests.selection collection
      // if the user dragged a Table to the myGuests diagram, cancel the drag
      if (e.subject.any(isTable)) {
        myDiagram.currentTool.doCancel();
        myGuests.currentTool.doCancel();
        return;
      }
      myDiagram.selection.each(function(n) {
```

```
      if (isPerson(n)) unassignSeat(n.data);
    });
    myDiagram.disableSelectionDeleted = true;
    myDiagram.commandHandler.deleteSelection();
    myDiagram.disableSelectionDeleted = false;
    myGuests.selection.each(function(n) {
      if (isPerson(n)) unassignSeat(n.data);
    });
  });

function isPerson(n) { return n !== null && n.category === ""; }

function isTable(n) { return n !== null && n.category !== ""; }

// Highlight the empty and occupied seats at a "Table" Node
function highlightSeats(node, show) {
  if (isPerson(node)) {  // refer to the person's table instead
    node = node.diagram.findNodeForKey(node.data.table);
    if (node === null) return;
  }
  var guests = node.data.guests;
  for (var sit = node.elements; sit.next();) {
    var seat = sit.value;
    if (seat.name) {
      var num = parseFloat(seat.name);
      if (isNaN(num)) continue;
      var seatshape = seat.findObject("SEATSHAPE");
      if (!seatshape) continue;
      if (show) {
        if (guests[seat.name]) {
          seatshape.stroke = "red";
        } else {
          seatshape.stroke = "green";
        }
      } else {
        seatshape.stroke = "white";
      }
    }
  }
}



function assignPeopleToSeats(node, coll, pt) {
  if (isPerson(node)) {  // refer to the person's table instead
    node = node.diagram.findNodeForKey(node.data.table);
    if (node === null) return;
  }
  if (coll.any(isTable)) {
    // if dragging a Table, don't allow it to be dropped onto another table
```

```
      myDiagram.currentTool.doCancel();
      return;
    }
    // OK -- all Nodes are people, call assignSeat on each person data
    coll.each(function(n) { assignSeat(node, n.data, pt); });
    positionPeopleAtSeats(node);
  }

  // Given a "Table" Node, assign one guest data to a seat at that table.
  // Also handles cases where the guest represents multiple people, because guest.plus >
0.
  // This tries to assign the unoccupied seat that is closest to the given point in
document coordinates.
  function assignSeat(node, guest, pt) {
    if (isPerson(node)) {  // refer to the person's table instead
      node = node.diagram.findNodeForKey(node.data.table);
      if (node === null) return;
    }
    if (guest instanceof go.GraphObject) throw Error("A guest object must not be a
GraphObject: " + guest.toString());
    if (!(pt instanceof go.Point)) pt = node.location;

    // in case the guest used to be assigned to a different seat, perhaps at a different
table
    unassignSeat(guest);

    var model = node.diagram.model;
    var guests = node.data.guests;
    // iterate over all seats in the Node to find one that is not occupied
    var seat= findClosestUnoccupiedSeat(node, pt);
    if (seat.name) {
      model.setDataProperty(guests, seat.name, guest.key);
      model.setDataProperty(guest, "table", node.data.key);
      model.setDataProperty(guest, "seat", parseFloat(seat.name));
      model.setDataProperty(guest, "color", seat.color);
    }

    var plus = guest.plus;
    if (plus) {  // represents several people
      // forget the "plus" info, since next we create N copies of the node/data
      guest.plus = undefined;
      model.updateTargetBindings(guest);
      for (var i = 0; i < plus; i++) {
        var copy = model.copyNodeData(guest);
        // don't copy the seat assignment of the first person
        copy.table = undefined;
        copy.seat = undefined;
        model.addNodeData(copy);
        assignSeat(node, copy, pt);
      }
    }
```

```
  }

  // Declare that the guest represented by the data is no longer assigned to a seat at a
table.
  // If the guest had been at a table, the guest is removed from the table's list of
guests.
  function unassignSeat(guest) {
    if (guest instanceof go.GraphObject) throw Error("A guest object must not be a
GraphObject: " + guest.toString());
    var model = myDiagram.model;
    // remove from any table that the guest is assigned to
    if (guest.table) {
      var table = model.findNodeDataForKey(guest.table);
      if (table) {
        var guests = table.guests;
        if (guests) model.setDataProperty(guests, guest.seat.toString(), undefined);
      }
    }
    model.setDataProperty(guest, "table", undefined);
    model.setDataProperty(guest, "seat", undefined);
    model.setDataProperty(guest, "color", undefined);
  }

  // Find the name of the unoccupied seat that is closest to the given Point.
  // This returns null if no seat is available at this table.
  function findClosestUnoccupiedSeat(node, pt) {
    if (isPerson(node)) {  // refer to the person's table instead
      node = node.diagram.findNodeForKey(node.data.table);
      if (node === null) return;
    }
    var guests = node.data.guests;
    var closestseatname = null;
    var seatcolor = null;
    var closestseatdist = Infinity;
    // iterate over all seats in the Node to find one that is not occupied
    for (var sit = node.elements; sit.next();) {
      var seat = sit.value;
      if (seat.name) {
        var num = parseFloat(seat.name);
        if (isNaN(num)) continue;  // not really a "seat"
        if (guests[seat.name]) continue;  // already assigned
        var seatloc = seat.getDocumentPoint(go.Spot.Center);
        var seatdist = seatloc.distanceSquaredPoint(pt);
        if (seatdist < closestseatdist) {
          closestseatdist = seatdist;
          closestseatname = seat.name;
          seatcolor=seat.data;
        }
      }
    }
    return {name: closestseatname, color: seatcolor};
```

```
  }

  // Position the nodes of all of the guests that are seated at this table
  // to be at their corresponding seat elements of the given "Table" Node.
  function positionPeopleAtSeats(node) {
    if (isPerson(node)) {  // refer to the person's table instead
      node = node.diagram.findNodeForKey(node.data.table);
      if (node === null) return;
    }
    var guests = node.data.guests;
    var model = node.diagram.model;
    for (var seatname in guests) {
      var guestkey = guests[seatname];
      var guestdata = model.findNodeDataForKey(guestkey);
      positionPersonAtSeat(guestdata);
    }
  }

  function positionPersonAtSeat(guest) {
    if (guest instanceof go.GraphObject) throw Error("A guest object must not be a
GraphObject: " + guest.toString());
    if (!guest || !guest.table || !guest.seat) return;
    var diagram = myDiagram;
    var table = diagram.findPartForKey(guest.table);
    var person = diagram.findPartForData(guest);
    if (table && person) {
      var seat = table.findObject(guest.seat.toString());
      var loc = seat.getDocumentPoint(go.Spot.Center);
      person.location = loc;
    }
  }
  </script>
</template>
```

client/views/layout.html

```
<template name='layout'>
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
  <div class="container">
    {{> sAlert}}
    {{> yield}}
  </div>
</template>
```

client/views/main.html

```
<head>
  <title>Castellers</title>
  <link rel="icon" type="image/jpg" href="imatges/favicon.jpg"/>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/gojs/1.6.2/go.js"></script>
</head>
```

client/views/pinya.html

```
<template name="pinya">
      <h1>Auto Pinyes <strong>{{currentUser.profile.colla}}</strong></h1>
      <button id="all" class="btn btn-primary">Tots els Castellers</button>
      <br>
      <br>
      {{> chart}}
</template>
```

client/views/preguntes.html

```
<template name="preguntes">
    <div class="row">
      <div class="col-xs-6">
        <h3><strong>Esdeveniment</strong></h3>
      </div>
      <div class="col-xs-2">
        <h3><strong>SI</strong></h3>
      </div>
      <div class="col-xs-2">
        <h3><strong>NO</strong></h3>
      </div>
      <div class="col-xs-2">
        <h3><strong>NSNC</strong></h3>
      </div>
    </div>
    <div style="font-size:150%">
    {{#each preguntes}}
      {{> pregunta}}
    {{/each}}
    <br>
    <button id="add-pregunta" class="btn btn-success">Nou esdeveniment</button>
  </div>
</template>


<template name="pregunta">
  <div class="row">
    <div class="col-xs-6">
      <a href="/capde/esdeveniments/{{_id}}">{{pregunta}}</a>
    </div>
```

```
    <div class="col-xs-2">
      {{si.length}}
    </div>
    <div class="col-xs-2">
      {{no.length}}
    </div>
    <div class="col-xs-2">
      {{nsnc.length}}
    </div>
  </div>
</template>

<template name="preguntaInfo">
  <a href="/capde/esdeveniments" class="btn btn-link">Cancelar</a>
  <h1>{{pregunta}} <button class="btn btn-danger" id="esborrar-
pregunta">Esborrar</button></h1>
  <button id="create-chart" class="btn btn-success">Crear Pinya</button>
  <div class="row">
    <div class="col-xs-4">
      <h3><strong>SI ({{si.length}})</strong></h3>
      <br>
      {{#each si}}
        <span style="font-size:150%"><a href="/capde/casteller/{{this}}">{{Nom
this}}</a></span>
        <br>
      {{/each}}
    </div>
    <div class="col-xs-4">
      <h3><strong>NO ({{no.length}})</strong></h3>
      <br>
      {{#each no}}
        <span style="font-size:150%"><a href="/capde/casteller/{{this}}">{{Nom
this}}</a></span>
        <br>
      {{/each}}
    </div>
    <div class="col-xs-4">
      <h3><strong>NSNC ({{nsnc.length}})</strong></h3>
      <br>
      {{#each nsnc}}
        <span style="font-size:150%"><a href="/capde/casteller/{{this}}">{{Nom
this}}</a></span>
        <br>
      {{/each}}
    </div>
  </div>
</template>


<template name="preguntaAnswer">
  <div class="row" style="background-color: {{getStyle}}"> <!--style="background-color:
```

```
#ffccca"  #d1f3d1-->
    <div class="col-xs-6">
      <h3>{{pregunta}}</h3>
    </div>
    <div class="radio">
      <div class="col-xs-3">
        <input type="radio" name="{{_id}}" id="{{_id}}" value="true" checked="{{isChecked
true}}">
      </div>
      <div class="col-xs-3">
        <input type="radio" name="{{_id}}" id="{{_id}}" value="false"
checked="{{isChecked false}}">
      </div>
    </div>
  </div>
</template>
```

client/views/register.html

```
<template name="register">
  <a href="/" class="btn btn-link">Cancelar</a>
  <h1>Registrar-se</h1>
  <div class="form-group">
    <label for="nom">Nom</label>
    <input type="text" class="form-control" id="nom" placeholder="Nom">
  </div>
  <div class="form-group">
    <label for="alçada">Alçada (cm)</label>
    <input type="number" class="form-control" id="alçada" placeholder="Alçada">
  </div>
  <div class="form-group">
    <label for="nom">Colla</label>
    <br>
    <select class="form-control" id="colla">
      {{#each colles}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>

  <div class="form-group">
    <label for="pos1">Posició 1:</label>
    <select class="form-control" id="pos1">
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="form-group">
    <label for="pos2">Posició 2:</label>
```

```
    <select class="form-control" id="pos2">
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>
  <div class="form-group">
    <label for="pos3">Posició 3:</label>
    <select class="form-control" id="pos3">
      {{#each posicions}}
        <option value={{this.nom}}>{{this.nom}}</option>
      {{/each}}
    </select>
  </div>


  <button class="btn btn-success">Registrar-se</button>
</template>
```

client/helpers/helpers.js

```
Template.preguntaAnswer.helpers({
  getStyle: function(){
    if(this.no.indexOf(Meteor.userId())>-1){
      return "#ffccca"
    }else if (this.si.indexOf(Meteor.userId())>-1){
      return "#d1f3d1"
    }else{
      return null
    }
  },
  isChecked:function(value){
    if (this.si.indexOf(Meteor.userId())>-1){
      return value;
    }else if (this.no.indexOf(Meteor.userId())>-1){
      return !value;
    }
  }
});

Template.main.helpers({
  test: function(){
    return 10;
  }
});

Template.listCastellers.helpers({
```

```
  castellers: function(){
    var user = Meteor.users.findOne({_id:Meteor.userId()});
    return Meteor.users.find(
      {
        "profile.colla" : user.profile.colla,
        "profile.admin" : false
      }, {sort:{"profile.nom":1}});
  }
});
```

client/hrlpers/registerHelpers.js

```
Template.registerHelper("colles", function(){
  return Colles.find();
});

Template.registerHelper("membres", function(){
  list=Session.get('membres');
  result=[];
  for(i=0;i<list.length;i++){
    user=Meteor.users.findOne({_id:list[i]});
    if(user){
      result.push({key:user.profile.nom})
    }
  }
  return result;
});

Template.registerHelper("Nom", function(userId){
  var user = Meteor.users.findOne({"_id":userId});
  return user.profile.nom;
});

Template.registerHelper("preguntes", function(){
  return Preguntes.find();
});

Template.registerHelper("itemsReady", function(){
  return Meteor.subscribe("preguntes").ready();
});

Template.registerHelper("posicions", function(){
  var posicions = [
    {nom:""},
    {nom:"Baix"},
    {nom:"Crossa"},
    {nom:"Agulla"},
    {nom:"Contrafort"},
    {nom:"Mans"},
```

```
    {nom:"Lateral"},
    {nom:"Vent"}
  ];
  return posicions;
});
```

client/helpers/sAlert.js

```
Meteor.startup(function () {
    sAlert.config({
        effect: 'slide',
        position: 'top-right',
        timeout: 3000,
        html: false,
        stack: true,
        offset: 0,
        beep: false,
        onClose: _.noop
    });

});
```

client/controllers/accountsController.js

```
Template.register.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    var name = template.find('#nom').value;
    var pos1 = template.find('#pos1').value;

    if(name==""){
      console.log(pos1);
      sAlert.error("Nom necessari!");
    }else{
      var colla = template.find('#colla').value;
      var alçada = template.find('#alçada').value;
      var pos1 = template.find('#pos1').value;
      var pos2 = template.find('#pos2').value;
      var pos3 = template.find('#pos3').value;
      createAccount(name,colla,alçada, pos1, pos2, pos3);
    };
  }
});

var createAccount = function(name,colla,alçada, pos1, pos2, pos3){
  Accounts.createUser({
    username : name+colla,
    password : colla,
    profile : {
```

```
      nom: name,
      colla : colla,
      alçada: alçada,
      admin:false,
      posicio1: pos1,
      posicio2: pos2,
      posicio3: pos3
    }
  },
  function(err){
    if (err){
      sAlert.error(err.reason);
    }else{
      Meteor.call("addCasteller", Meteor.userId(), colla, function(error,result){
        if (error){
          sAlert.error(error.reason);
        }
      });
    }
  });
  Router.go('/');
};

Template.login.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    username=template.find('#nom').value+template.find('#colla').value;
    password=template.find('#colla').value;
    Meteor.loginWithPassword(username,password,
      function(err){
        if (err){
          sAlert.error(err.reason);
        }
      }
    )
  }
});

Template.capdeLogin.events({
  'click button' : function(evt,template){
    evt.preventDefault();
    username="admin"+template.find('#colla').value;
    password=template.find('#password').value;
    Meteor.loginWithPassword(username,password,
      function(err){
        if (err){
          sAlert.error(err.reason);
        }
      }
    )
  }
```

```
});

Template.capdecollaLayout.events({
  'click #back' : function(evt,template){
    evt.preventDefault();
    Meteor.logout(function(err){
      if (err){
        sAlert.error(err.reason)
      }else{
        Router.go('/');
      }
    })
  }
});

Template.casteller.events({
  'click #back' : function(evt,template){
    evt.preventDefault();
    Meteor.logout(function(err){
      if (err){
        sAlert.error(err.reason)
      }else{
        Router.go('/');
      }
    })
  }
});
```

client/controllers/capdeController.js

```
Template.buttonGroup2.events({
  "change": function(event, template){
     if (event.target.value==1){
       Session.set("Position", "Baix");
     }else if (event.target.value==2){
       Session.set("Position", "Agulla");
     }else if (event.target.value==3){
       Session.set("Position", "Crossa");
     }else if (event.target.value==4){
       Session.set("Position", "Contrafort");
     }else if (event.target.value==5){
       Session.set("Position", "Mans");
     }else if (event.target.value==6){
       Session.set("Position", "Vent");
     }else if (event.target.value==7){
       Session.set("Position", "Lateral");
     }else{
       Session.set("Position", "");
     }
  }
```

```
});

Template.buttonGroup2.onRendered = function(){
  Session.setDefault("Position", "Baix");
}
```

client/controllers/castellersController.js

```
var limit = 1;

Template.casteller.events({
  "switchChange.bootstrapSwitch #checkbox": function (event, template) {
    var state = template.find("#checkbox").checked;
    var user=Meteor.users.findOne({_id:Meteor.userId()});
    if (user){
      user.profile.state=state;
      Meteor.users.update({_id:Meteor.userId()}, {$set:{profile:user.profile}});
    }
  },
  "click #save": function(event, template){
    preguntes = Preguntes.find().forEach(function(pregunta){
      var element= template.find('input:radio[name='+pregunta._id+']:checked');
      Meteor.call("updatePregunta", $(element)[0].id, $(element).val())
    })
    sAlert.success("Respostes guardades")
  }
});

Template.edit.events({
  "click button": function(event, template){

    var user=Meteor.users.findOne({_id:Meteor.userId()});
    if (user){
      var alçada = template.find('#alçada').value;
      var pos1 = template.find('#pos1').value;
      var pos2 = template.find('#pos2').value;
      var pos3 = template.find('#pos3').value;
      user.profile.alçada=alçada;
      user.profile.posicio1=pos1;
      user.profile.posicio2=pos2;
      user.profile.posicio3=pos3;

      Meteor.users.update({_id:Meteor.userId()}, {$set:{profile:user.profile}});
    }
    Router.go('/')
  }
});
```

```
Template.capdeEdit.events({
  "click button": function(event, template){
    var user=Meteor.users.findOne({_id:template.data._id});
    if (user){
      var alçada = template.find('#alçada').value;
      var pos1 = template.find('#pos1').value;
      var pos2 = template.find('#pos2').value;
      var pos3 = template.find('#pos3').value;
      user.profile.alçada=alçada;
      user.profile.posicio1=pos1;
      user.profile.posicio2=pos2;
      user.profile.posicio3=pos3;


      Meteor.users.update({_id:template.data._id},
{$set:{profile:user.profile}},function(err,result){
        if(err){
          sAlert.error(err);
        }else{
          Router.go('/capde/castellers')
        }
      });
    }
  }
});

Template.navBar.events({
  "click #checkbox": function(event, template){
    collaAccount=Meteor.users.findOne({_id:Meteor.userId()});
    if (collaAccount){
      colla = collaAccount.profile.colla;
      var array = getChecked(template);
      if (array.length>limit) {
        event.currentTarget.checked=false;
        sAlert.error("Màxim nombre de baixos")
      }else{
        user = Meteor.users.findOne({username:event.currentTarget.value+colla});
        Meteor.users.update({_id:user._id},
{$set:{"profile.baix":event.currentTarget.checked}});
      }
    }
  }
});

var getChecked=function(template){
  var selected = template.findAll( "input[type=checkbox]:checked");
  var array = _.map(selected, function(item) {
    return item.defaultValue;
  });
  return array;
}
```

```
Template.buttonGroup.events({
  "click": function(event, template){
    cleanChecboxes();
    limit = event.toElement.childNodes[1].value;
  }
});

var cleanChecboxes = function(){
  Meteor.users.find({}).forEach(
    function(casteller){
      Meteor.users.update({_id:casteller._id}, {$set:{
        "profile.baix": false
      }});
    }
  );
}
```

client/controllers/pinyaController.js

```
Template.pinya.events({
  "click #all": function(event, template){
    var user = Meteor.users.findOne({_id:Meteor.userId()});
    result = Meteor.users.find(
      {
        "profile.colla" : user.profile.colla,
        "profile.admin" : false
      }, {sort:{"profile.nom":1}});
      result=result.map(function(user){return user._id});
     Session.set("membres", result);
     Router.current().render(Template.pinya);
  }
});
```

client/controllers/preguntesController.js

```
Template.preguntes.events({
  "click #add-pregunta": function(event, template){
    swal({
      title: "Nou esdeveniment!",
      type: "input",
      showCancelButton: true,
      closeOnConfirm: false,
      animation: "slide-from-top",
      inputPlaceholder: "Assaig / Actuació / ..."
    }, function(inputValue){
      if (inputValue === false)
        return false;
      if (inputValue === "") {
```

```
        swal.showInputError("Error!");
        return false
      }
      Meteor.call("addPregunta", inputValue, membres(), function(error, result){
        if(error){
          swal("Error!",error,"error");
        }else{
          swal(inputValue + " creat!","","success");

        }
      });

    });
  }
});


Template.preguntaInfo.events({
  "click #esborrar-pregunta": function(event, template){
    id = this._id;
    swal({
      title: "Estàs segur?",
      text: "No podràs recuperar aquest esdeveniment!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Si!",
      cancelButtonText: "No",
      closeOnConfirm: false
    }, function(){
      Meteor.call("deletePregunta", id, function(error, result){
        if(error){
          swal("Error!",error,"error");
        }else{
         swal("Esdeveniment esborrat!","","success");
        };
      });
      history.back();
    });
  },
  "click #create-chart":function(event, template){
    Session.set("membres", template.data.si);
    Router.go('/capde/pinya')
  }
});

Template.preguntaAnswer.events({
  "click #optionsRadios": function(event, template){
     if (event.toElement.value){
       //template.find('#row').style.background-color="#ffccca";
```

```
    }
  }
});

function membres() {
  var user = Meteor.users.findOne({_id:Meteor.userId()});
  return Meteor.users.find(
    {
      "profile.colla" : user.profile.colla,
      "profile.admin" : false
    }
  ).map(function(u) {
    return u._id;
  });
}
```

A.5. Codi de la plataforma Quality Engine

server/startup.js

```
Meteor.startup(function(){
  Meteor.users.allow({
    update: function(userId, doc){
      if (Meteor.users.findOne({_id:userId, "profile.admin":true})){
        return true;
      }else if(doc._id == userId){
        return true;
      }else{
        return false;
      }
    },
    remove: function(userId){
      return (Meteor.users.findOne({_id:userId, "profile.admin":true}));
    }
  });

  if (Meteor.users.find().count() === 0){
    var userObject = {
      username: "admin",
      password: "admin",
      profile:{
        admin:true
      }
    };
    Accounts.createUser(userObject);
  }
});
```

models/years.js

```
Years = new Mongo.Collection("years");

if(Meteor.isServer){
  Meteor.publish("years", function(){
    return Years.find();
  });

  Meteor.methods({
    addYear:function(year){
      Years.insert({year:year});
    },
    deleteYear:function(year){
      Years.remove({year:year})
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("years");
}
```

models/users.js

```
if(Meteor.isServer){
  Meteor.publish("users", function(){
    return Meteor.users.find();
  });
  Meteor.methods({
    newUser:function(userObject){
      Accounts.createUser(userObject);
    },
    changePsw:function(id, password){
      Accounts.setPassword(id, password);
    }
  })
}

if(Meteor.isClient){
  Meteor.subscribe("users");
}
```

models/types.js

```
Types = new Mongo.Collection("types");

if(Meteor.isServer){
  Meteor.publish("types", function(){
    return Types.find({});
  });

  Meteor.methods({
    addType:function(name){
      resultInsert = Types.insert({name:name});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Types.insert", Types.findOne({_id:resultInsert}));

    },
    deleteType:function(Id){
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Types.remove", Types.findOne({_id:Id}));

      Types.remove({_id:Id});
    },
    updateType:function(Id,name){
      Types.update({_id:Id}, {$set:{name:name}});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Types.update", Types.findOne({_id:Id}));

    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("types");
}
```

models/tollgates.js

```
Tollgates = new Mongo.Collection("tollgates");

if(Meteor.isServer){
  Meteor.publish("tollgates", function(){
    return Tollgates.find();
  });
  Meteor.methods({
    addTollgate:function(releaseId, phase, date,link,result){
      rel=Releases.findOne({_id:releaseId});
      if(rel){
        proj=Projects.findOne({_id:rel.project});
        if (proj){
          scheme=Schemes.findOne({_id:proj.scheme});
```

```
          if(scheme){

            Meteor.call("copyChecklistTemplate", scheme.checklists[phase],
function(error, callResult){
              if(error){
                console.log(error);
              }else{
                resultInsert = Tollgates.insert({
                  release:releaseId,
                  phase:phase,
                  date:date,
                  link:link,
                  checklist:callResult,
                  result:result
                });

                Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Tollgates.insert", Tollgates.findOne({_id:resultInsert}));
              }
            });

            date=date.split('/')[2]
            year=Years.findOne({year:date});
            if (!year){
              Meteor.call("addYear", date);
            }
          }else{
            resultInsert = Tollgates.insert({
              release:releaseId,
              phase:phase,
              date:date,
              link:link,
              result:result
            });
            Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Tollgates.insert", Tollgates.findOne({_id:resultInsert}));

            date=date.split('/')[2]
            year=Years.findOne({year:date});
            if (!year){
              Meteor.call("addYear", date);
            }
          }
        }
      }
    },
    updateTollgate:function(tollId, date, link, result){
      var toll=Tollgates.findOne({_id:tollId});
      if(toll){
        if(toll.checklist){
          Tollgates.update({_id:toll._id}, {$set:{
```

```
              date:date,
              link:link,
              result:result
          }});
          Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Tollgates.update", Tollgates.findOne({_id:toll._id}));
        }else{
          rel=Releases.findOne({_id:toll.release});
          if(rel){
            proj=Projects.findOne({_id:rel.project});
            if (proj){
              scheme=Schemes.findOne({_id:proj.scheme});
              if(scheme){
                Meteor.call("copyChecklistTemplate", scheme.checklists[toll.phase],
function(error, callResult){
                  if(error){
                    console.log(error);
                  }else{
                    Tollgates.update({_id:toll._id}, {$set:{
                      date:date,
                      link:link,
                      result:result,
                      checklist:callResult
                    }});
                    Meteor.call("log",
Meteor.users.findOne({_id:Meteor.userId()}).username, "Tollgates.update",
Tollgates.findOne({_id:toll._id}));
                  }
                });
              }
            }
          }
        }
      },
      deleteTollgate:function(tollId){
        toll = Tollgates.findOne({_id:tollId});

        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Tollgates.remove", Tollgates.findOne({_id:tollId}));

        Tollgates.remove({_id:tollId});
        if(toll){
          Checklists.remove({_id:toll.checklist});

          year=toll.date.split('/')[2];
          toll2=Tollgates.findOne({date:{$regex:".*"+year}});
          if (!toll2){
            project = Projects.findOne({createdAt:year});
            if (!project){
              Meteor.call("deleteYear", year);
```

```
          }
        }
      }
    }
  });
}


if(Meteor.isClient){
  Meteor.subscribe("tollgates");
}
```

models/targets.js

```
Targets = new Mongo.Collection("targets");
Categories = new Mongo.Collection("categories");

if(Meteor.isServer){
  Meteor.publish("targets", function(){
    return Targets.find({});
  });
  Meteor.publish("categories", function(){
    return Categories.find({});
  });

  Meteor.methods({
    addTarget:function(name){
      resultInsert = Targets.insert({name:name});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Targets.insert", Targets.findOne({_id:resultInsert}));

    },
    deleteTarget:function(Id){
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Targets.remove", Targets.findOne({_id:Id}));

      Targets.remove({_id:Id});
    },
    updateTarget:function(Id,name){
      Targets.update({_id:Id}, {$set:{name:name}});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Targets.update", Targets.findOne({_id:Id}));

    },



    addCategory:function(name){
```

```
      resultInsert = Categories.insert({name:name});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Categories.insert", Categories.findOne({_id:resultInsert}));

    },
    deleteCategory:function(Id){
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Categories.remove", Categories.findOne({_id:Id}));

      Categories.remove({_id:Id});
    },
    updateCategory:function(Id,name){
      Categories.update({_id:Id}, {$set:{name:name}});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Categories.update", Categories.findOne({_id:Id}));

    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("targets");
  Meteor.subscribe("categories");
}
```

models/releasesTime.js

```
ReleasesTime = new Mongo.Collection("releasesTime");

if(Meteor.isServer){
  Meteor.publish("releasesTime", function(){
    return ReleasesTime.find({});
  });

  Meteor.methods({
    addReleaseTime:function(phase, name, start, end, project){
      ReleasesTime.insert({
        phase:phase,
        name:name,
        start: start,
        end: end,
        project:project
      });
    },
    deleteReleaseTime:function(Id){
      ReleasesTime.remove({_id:Id});
    },
    updateReleaseTime:function(Id,phase, name, start, end){
      ReleasesTime.update({_id:Id}, {$set:{
```

```
        name:name,
        start:start,
        end:end,
        phase:phase
      }});
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("releasesTime");
}
```

models/releases.js

```
Schemes = new Mongo.Collection("schemes");

if(Meteor.isServer){
  Meteor.publish("schemes", function(){
    return Schemes.find();
  });
  Meteor.methods({
    createScheme:function(name, phase3FL, phase3CL, phase4FL, phase4CL, phase5FL,
phase5CL, phase6FL){
      Schemes.insert({
        name:name,
        checklists:{
          phase3FL:phase3FL,
          phase3CL:phase3CL,
          phase4FL:phase4FL,
          phase4CL:phase4CL,
          phase4FL:phase5FL,
          phase5CL:phase5CL,
          phase5FL:phase6FL
        }
      });
    },
    updateScheme:function(Id, name, phase3FL, phase3CL, phase4FL, phase4CL, phase5FL,
phase5CL, phase6FL, phase6CL){
      Schemes.update({_id:Id}, {$set:{
        name:name,
        checklists:{
          phase3FL:phase3FL,
          phase3CL:phase3CL,
          phase4FL:phase4FL,
          phase4CL:phase4CL,
          phase4FL:phase5FL,
          phase5CL:phase5CL,
          phase6FL:phase6FL,
          phase6CL:phase6CL
```

```
        }
      }});
    },
    deleteScheme:function(Id){
      Schemes.remove({_id:Id});
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("schemes");
}
```

models/projects.js

```
Projects = new Mongo.Collection("projects");

if(Meteor.isServer){
  Meteor.publish("projects", function(){
    return Projects.find();
    });

    Meteor.methods({
      createProject:function(name, ref, projectLead, sw, hw, mech, system, QA, TV, TR,
compliance, scheme){
        date = new Date().getFullYear().toString();
        resultInsert = Projects.insert({
          _id:ref,
          name:name,
          projectLead:projectLead,
          swLead:sw,
          hwLead:hw,
          mechLead:mech,
          system:system,
          QA:QA,
          TV:TV,
          TR:TR,
          compliance:compliance,
          scheme:scheme,
          createdAt:date,
          archived:false
        });
        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Projects.insert", Projects.findOne({_id:resultInsert}));

        year=Years.findOne({year:date});
        if (!year){
          Meteor.call("addYear", date);
        }
      },
```

```
      deleteProject:function(Id){
        project=Projects.findOne({_id:Id});

        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Projects.remove", Projects.findOne({_id:Id}));

        Projects.remove({_id:Id});
        Releases.find({project:Id}).forEach(function(release){
          Meteor.call("deleteRelease", release._id);
        });
        KPIs.find({project:Id}).forEach(function(kpi){
          Meteor.call("deleteKPI", kpi._id);
        });
        if(project){
          year=project.createdAt;
          toll=Tollgates.findOne({date:{$regex:".*"+year}});
          if (!toll){
            project = Projects.findOne({createdAt:year});
            if (!project){
              Meteor.call("deleteYear", year);
            }
          }
        }
      },
      updateProject:function(id, projectLead, sw, hw, mech, system, QA, TV, TR,
compliance, scheme){
        Projects.update({_id:id}, {$set:{
          projectLead:projectLead,
          swLead:sw,
          hwLead:hw,
          mechLead:mech,
          system:system,
          QA:QA,
          TV:TV,
          TR:TR,
          compliance:compliance,
          scheme:scheme
        }});
        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Projects.update", Projects.findOne({_id:id}));

      },
      openProject:function(id){
        Projects.update({_id:id}, {$set:{archived:false}});
        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Projects.update", Projects.findOne({_id:id}));

      },
      closeProject:function(id){
        Projects.update({_id:id}, {$set:{archived:true}});
        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
```

```
"Projects.update", Projects.findOne({_id:id}));

      },
      checkProject:function(id){
        if (Releases.find({project:id,
archived:true}).count()==Releases.find({project:id}).count()){
          Projects.update({_id:id}, {$set:{archived:true}});
        }
      }
    });
}

if(Meteor.isClient){
  Meteor.subscribe("projects");
}
```

models/phaseTime.js

```
Phases = new Mongo.Collection("phases");

if(Meteor.isServer){
  Meteor.publish("phases", function(){
    return Phases.find({});
  });

  Meteor.methods({
    addPhase:function(project, name, start, end, open){
      Phases.insert({
        project:project,
        name:name,
        start: start,
        end: end,
        open:open
      });
    },
    deletePhase:function(Id){
      Phases.remove({_id:Id});
    },
    updatePhase:function(Id, name, start, end, open){
      Phases.update({_id:Id}, {$set:{
        name:name,
        start:start,
        end:end,
        open:open
      }});
    }
  });
}
```

```
if(Meteor.isClient){
  Meteor.subscribe("phases");
}
```

models/phaseRelease.js

```
PhaseReleases = new Mongo.Collection("phase-releases");

if(Meteor.isServer){
  Meteor.publish("phase-releases", function(){
    return PhaseReleases.find();
  });

  Meteor.methods({
    addPhaseRelease:function(phase, name){
      PhaseReleases.insert({
        phase:phase,
        name:name
      });
    },
    deletePhaseRelease:function(relId){
      PhaseReleases.remove({_id:relId})
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("phase-releases");
}
```

models/offers.js

```
Offers = new Mongo.Collection("offers");

if(Meteor.isServer){
  Meteor.publish("offers", function(){
    return Offers.find({});
  });
  Meteor.methods({
    addOffer:function(customer, bu, product, scope, status, svn, comments){
      Offers.insert({
        customer:customer,
        bu:bu,
        product:product,
        scope:scope,
        status:status,
        svn:svn,
        comments:comments,
        archived:false
```

```
      })
    },
    deleteOffer:function(id){
      Offers.remove({_id:id});
    },
    updateOffer:function(id, customer, bu, product, scope, status, svn, comments){
      Offers.update({_id:id}, {$set:{
        customer:customer,
        bu:bu,
        product:product,
        scope:scope,
        status:status,
        svn:svn,
        comments:comments
      }});
    },
    updateStatusOffer:function(id, value){
      if(value!="Offer"){
        Offers.update({_id:id}, {$set:{status:value, archived:true}});
      }else{
        Offers.update({_id:id}, {$set:{status:value}});
      }
    },
    updateCommentOffer:function(id, value){
      Offers.update({_id:id}, {$set:{comments:value}});
    },
    switchArchiveOffer:function(id){
      offer=Offers.findOne({_id:id});
      if(offer){
        console.log(offer.archieved);

        Offers.update({_id:id}, {$set:{archived:(!offer.archived)}});
      }
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("offers");
}
```

models/logs.js

```
Meteor.methods({
  log:function(user, method, dataMethod){
    var fs = Npm.require('fs');
    var path = Npm.require('path');
    var basePath = path.resolve('.').split('.meteor')[0] + "logs.txt";

    data = "[{timestamp: "+ new Date().toISOString()+"}, {user: "+user+"}, {method:
```

```
"+method+"}, {data: "+JSON.stringify(dataMethod)+"}]\n";
    fs.appendFile(basePath, data, function(err) {
      if(err){
        console.log(err);
      }
    });
  }
})
```

models/KPIs.js

```
KPIs = new Mongo.Collection("kpis");

if(Meteor.isServer){
  Meteor.publish("kpis", function(){
    return KPIs.find({});
  });

  Meteor.methods({
    addKPI:function(kpi){
      resultInsert = KPIs.insert(kpi);
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"KPIs.insert", KPIs.findOne({_id:resultInsert}));

    },
    updateKPI:function(id, kpi){
      KPIs.update({_id:id}, {$set:kpi});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"KPIs.update", KPIs.findOne({_id:id}));

    },
    deleteKPI:function(id){
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"KPIs.remove", KPIs.findOne({_id:id}));

      KPIs.remove({_id:id});
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("kpis");
}
```

models/eventsTime.js

```
Events = new Mongo.Collection("events");

if(Meteor.isServer){
  Meteor.publish("events", function(){
    return Events.find({});
  });

  Meteor.methods({
    addEvent:function(name, date, rel,phase, result, category, project, type){
      Events.insert({
        name:name,
        date: date,
        release: rel,
        result:result,
        category:category,
        project: project,
        type:type
      });
    },
    deleteEvent:function(Id){
      Events.remove({_id:Id});
    },
    updateEvent:function(Id, name, date,phase, rel, result, category, type){
      Events.update({_id:Id}, {$set:{
        name:name,
        date: date,
        phase:phase,
        release: rel,
        result:result,
        category:category,
        type:type
      }});
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("events");
}
```

models/checksGroup.js

```
checksGroups = new Mongo.Collection("checksGroups");

if(Meteor.isServer){
  Meteor.publish("checksGroups", function(){
    return checksGroups.find({});
  });
```

```
  Meteor.methods({
    addGroup:function(Id,name){
       checksGroups.insert({_id:Id,name:name});
    },
    deleteGroup:function(Id){
      checksGroups.remove({_id:Id});
      Checks.remove({type:Id});
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("checksGroups");
}
```

models/checks.js

```
Checks = new Mongo.Collection("checks");

if(Meteor.isServer){
  Meteor.publish("checks", function(){
    return Checks.find();
  });
  Meteor.methods({
    createCheck:function(id, desc){
      Checks.insert({_id:id, desc:desc,type:id.split(".")[0]});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Checks.insert", Checks.findOne({_id:id}));

    },
    editCheck:function(id,desc){
      Checks.update({_id:id}, {$set:{
        desc:desc
      }});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Checks.update", Checks.findOne({_id:id}));

      Meteor.call("checkUpdated", id);
    },
    deleteCheck:function(id){
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Checks.remove", Checks.findOne({_id:id}));

      Checks.remove({_id:id});
    }
  });
}
```

```
if(Meteor.isClient){
  Meteor.subscribe("checks");
}
```

models/checklists.js

```
Checklists = new Mongo.Collection("checklists");

if(Meteor.isServer){
  Meteor.publish("checklists", function(){
    return Checklists.find();
  });
  Meteor.methods({
    createChecklist:function(name, phase, selected){
      var types=[];
      for(i=0;i<selected.length;i++){
        check=Checks.findOne({_id:selected[i]});
        if(check){
          selected[i]={
            _id:check._id,
            desc:check.desc,
            type:check.type,
            applicable:true,
            result:"",
            comments:"",
            criticality:""
          };
          if(types.indexOf(check.type)<0){
            types.push(check.type);
          }
        }
      }
      value = Checklists.insert({
        name:name,
         template:true,
         phase:phase,
         assistants:"",
         conclusions:"",
         types:types,
         checks:selected
        });
        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Checklists.insert", Checklists.findOne({_id:value}));

    },
    checkUpdated:function(checkId){
      var check=Checks.findOne({_id:checkId});
      list=Checklists.find({template:true, checks:{$elemMatch:{_id:checkId}}});
      list.forEach(function(element){
        for(i=0;i<element.checks.length;i++){
```

```
          if(element.checks[i]._id==check._id){
            element.checks[i].desc=check.desc;
            Checklists.update({_id:element._id}, {$set:{
              checks:element.checks
            }});
          }
        }
      });
    },
    updateChecklist:function(Id, name, phase,selected){
      list=Checklists.findOne({_id:Id});
      if(list){
        var types=list.types;
        for(i=0;i<selected.length;i++){
          check=Checks.findOne({_id:selected[i]});
          if(check){
            selected[i]={
              _id:check._id,
              desc:check.desc,
              type:check.type,
              applicable:true,
              result:"",
              comments:"",
              criticality:""
            }
            if(types.indexOf(check.type)<0){
              types.push(check.type);
            }
          }
        };
        Checklists.update({_id:Id}, {$set:{
          name:name,
          template:true,
          phase:phase,
          assistants:"",
          conclusions:"",
          checks:selected,
          types:types
        }});
        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Checklists.update", Checklists.findOne({_id:Id}));

      }
    },
    copyChecklistTemplate:function(templateId){
      checklist=Checklists.findOne({_id:templateId});
      if(checklist){
        checklist.template=false;
        checklist._id=Checklists._makeNewID();
        value = Checklists.insert(checklist);
        Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
```

```
"Checklists.insert", Checklists.findOne({_id:value}));
         return value;
      }else{
        return "";
      }
    },
    changeChecklist:function(checklist){
      Checklists.update({_id:checklist._id}, {$set:{
        assistants:checklist.assistants,
        conclusions:checklist.conclusions,
        checks:checklist.checks
      }});
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Checklists.update", Checklists.findOne({_id:checklist._id}));

    },
    deleteChecklist:function(Id){
      Meteor.call("log", Meteor.users.findOne({_id:Meteor.userId()}).username,
"Checklists.delete", Checklists.findOne({_id:Id}));

      Checklists.remove({_id:Id});
    }
  });
}

if(Meteor.isClient){
  Meteor.subscribe("checklists");
}
```

lib/router.js

```
Router.configure({
  layoutTemplate: 'layout'
});

Router.map(function(){
  this.route('home',{path:'/'});
  this.route('createUser',{path:'/users/new'});
  this.route('usersLogIn',{path:'/login'});
  this.route('usersList',{path:'/users'});
  this.route('createProject',{path:'/project/new'});
  this.route('userInfo', {path:'/users/:_id',
    data:function(){
      return Meteor.users.findOne({_id:this.params._id});
    }
  });
  this.route('projectPage', {path:'/project/:ref',
    data:function(){
      return Projects.findOne({_id:this.params.ref});
    }
```

```
});
this.route('timeline', {path:'/project/:ref/timeline',
  data:function(){
    return Projects.findOne({_id:this.params.ref});
  }
});
this.route('newPhase', {path:'/project/:ref/phases/new',
  data:function(){
    return Projects.findOne({_id:this.params.ref});
  }
});
this.route('newReleaseTime', {path:'/project/:ref/releases/new',
  data:function(){
    return Projects.findOne({_id:this.params.ref});
  }
});
this.route('newEvent', {path:'/project/:ref/events/new',
  data:function(){
    return Projects.findOne({_id:this.params.ref});
  }
});
this.route('projectEdit', {path:'/project/:ref/edit',
  data:function(){
    return Projects.findOne({_id:this.params.ref});
  }
});
this.route('tollgateInfo', {path:'/:relId/tollgate/:tollId',
  data:function(){
    return Tollgates.findOne({_id:this.params.tollId});
  }
});
this.route('newTollgate', {path:'/:relId/tollgate/new/:phase',
  data:function(){
    return {release:this.params.relId, phase:this.params.phase};
  }
});

this.route('schemesList',{path:'/checklist/schemes'});
this.route('newScheme',{path:'/checklist/schemes/new'});
this.route('schemeEdit', {path:'/checklist/schemes/:id',
  data:function(){
    return Schemes.findOne({_id:this.params.id});;
  }
});

this.route('checklistList',{path:'/checklist/templates'});
this.route('newChecklist',{path:'/checklist/templates/new'});
this.route('checklistEdit', {path:'/checklist/templates/:id',
  data:function(){
    return Checklists.findOne({_id:this.params.id});;
  }
```

```
});
this.route('checklistInfo', {path:'/checklist/:id',
  data:function(){
    return Checklists.findOne({_id:this.params.id});;
  }
});

this.route('checksList',{path:'/checks'});
this.route('checksInfo', {path:'/checks/:id',
  data:function(){
    return Checks.findOne({_id:this.params.id});;
  }
});
this.route('checksEdit', {path:'/checks/:id/edit',
  data:function(){
    return Checks.findOne({_id:this.params.id});;
  }
});

this.route('KPIs',{path:'/KPIs'});
this.route('targets',{path:'/KPIs/targets'});
this.route('newKPI',{path:'/project/:Id/KPIs/new',
  data:function(){
    return Projects.findOne({_id:this.params.Id});;
  }
});
this.route('editKPI',{path:'/project/:Id/KPIs/:kpiID',
  data:function(){
    return KPIs.findOne({_id:this.params.kpiID});
  }
});

this.route('offers',{path:'/offers'});
this.route('newOffer',{path:'/offers/new'});
this.route('offerInfo', {path:'/offers/:id',
  data:function(){
    return Offers.findOne({_id:this.params.id});;
  }
});

this.route('categories',{path:'/timeline/categories'});
this.route('types',{path:'/events/types'});

this.route('timelineInfo', {path:'/project/:id/timeline/:event',
  data:function(){
    data = Phases.findOne({_id:this.params.event, project:this.params.id});
    if(data){
      return {data:data, type:"Phase"};
    }else{
      data = ReleasesTime.findOne({_id:this.params.event, project:this.params.id});
```

```
        if(data){
          return {data:data, type:"Release"};
        }else{
          data = Events.findOne({_id:this.params.event, project:this.params.id});
          if(data){
            return {data:data, type:"Event"}
          }
        }
      }
    }
  });
  this.route('timelineEdit', {path:'/project/:id/timeline/:event/edit',
    data:function(){
      data = Phases.findOne({_id:this.params.event, project:this.params.id});
      if(data){
        return {data:data, type:"Phase"};
      }else{
        data = ReleasesTime.findOne({_id:this.params.event, project:this.params.id});
        if(data){
          return {data:data, type:"Release"};
        }else{
          data = Events.findOne({_id:this.params.event, project:this.params.id});
          if(data){
            return {data:data, type:"Event"}
          }
        }
      }
    }
  })

});

if(Meteor.isClient){
  Session.setDefault('hide', []);
  Session.setDefault('showAll', false);
  date= new Date().getFullYear().toString();
  Session.setDefault('year', date);
  Session.setDefault('tollgates', false);
  Session.setDefault('closedProjects', false);
  Session.setDefault('archivedOffers', false);
  Session.setDefault("phaseEvent", "");

}
```

client/views/css/css.css

```
.table td {
   text-align: center;
}
.table th {
```

```
   text-align: center;
}

.no-border {
    border: 0;
    box-shadow: none;
}

td a {
    display:block;
    width:100%;
}

a:hover, a:focus {
    text-decoration: inherit;;
}

.form-control[disabled], fieldset[disabled] .form-control {
    cursor: default;
}

input[type="radio"][disabled], input[type="checkbox"][disabled],
input[type="radio"].disabled, input[type="checkbox"].disabled, fieldset[disabled]
input[type="radio"], fieldset[disabled] input[type="checkbox"] {
    cursor: default;
}

.btn[disabled]{
  cursor:default;
}




*
{
    padding: 0;
    margin: 0;
}

body
{
    height: 100%;
    width: 100%;
}
table
{
    border-collapse: collapse;
}
```

```
.outer-container
{
    position: relative;
    top:0;
    left: 0;
    right: 300px;
    bottom: 40px;
    width: 1400px;
    height: 240px;
}

.inner-container
{
    height: 100%;
    overflow: hidden;
}

.table-body
{
    overflow: auto;
    height: 166px;
}

.table-header
{
    position: relative;
}

.header-cell
{
    text-align: left;
    height: 40px;
}
.body-cell
{
    text-align: left;
}
.col1, .col3, .col4, .col5
{
    width:120px;
    min-width: 120px;
}
.col2
{
    min-width: 300px;
}
```

client/views/categories.html

```
<template name="categories">
  <h1>Categories
    <button class="btn btn-success" id="new">
      <span class="glyphicon glyphicon-plus" aria-hidden="true" ></span> Create New
Category
    </button>
  </h1>
  <br>
  {{#each categories}}
    {{> categoryInfo}}
  {{/each}}
</template>

<template name="categoryInfo">
  <div class="row">
    <div class="col-xs-3">
      <h4>{{name}}</h4>
    </div>
    <div class="col-xs-2">
      <button type="button" class="btn btn-default" id="edit">Edit</button>
    </div>
    <div class="col-xs-2">
      <button type="button" class="btn btn-danger" id="delete">Delete</button>
    </div>
  </div>
</template>
```

client/views/chart.html

```
<template name="chart">
  <br>
  <div id={{this}}></div>
</template>
```

client/views/checklists.html

```
<template name="checklistList">
  <h1>Checklist Templates</h1>
  <a href="/checklist/templates/new" class="btn btn-success">
    <span class="glyphicon glyphicon-plus"></span>
    New Template
  </a>
  <br>
  <br>
  <h3></h3>
  {{#each phases}}
    <h2>{{name}}</h2>
    {{#each templates Id}}
```

```
      <h3><a href="/checklist/templates/{{_id}}">{{name}}</a></h3>
    {{/each}}
  {{/each}}
</template>

<template name="newChecklist">
  <h1>New Checklist Template</h1>
  <div class="form-group">
    <label for="phase">Name<span style="color:red">*</span></label>
    <input type="text" id="name" class="form-control" placeholder="Checklist Template
Name">
  </div>
  <div class="form-group">
    <label for="phase">Phase</label>
    <select class="form-control" id="phase">
      {{#each phases}}
        <option value={{Id}}>{{name}}</option>
      {{/each}}
    </select>
  </div>
  <h3>Checks</h3>
  {{#each groups}}
    <h4>{{_id}}. {{name}}
      <button class="btn btn-success" id={{_id}}>
        <span class="glyphicon glyphicon-plus"></span>
        New {{name}} Check
      </button>
    </h4>
    {{#each checks _id}}
      <div class="checkbox">
        <label><input type="checkbox" value="{{_id}}">{{_id}} - {{desc}}</label>
      </div>
    {{/each}}
  {{/each}}
  <span style="color:red">*Required</span>
  <br>
  <br>
  <button type="button" id="save" class="btn btn-success btn-lg">
    <span class="glyphicon glyphicon-saved"></span>
    Save
  </button>
  <button type="button" id="cancel" class="btn btn-link">Cancel</button>
</template>

<template name="checklistEdit">
  <h1>Checklist Template: {{name}}
    <button type="button" class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
    </button>
  </h1>
  <div class="form-group">
```

```
    <label for="phase">Name</label>
    <input type="text" id="name" class="form-control" placeholder="Checklist Template
Name" value={{name}}>
  </div>
  <div class="form-group">
    <label for="phase">Phase</label>
    <select class="form-control" id="phase">
      {{#each phases}}
        <option value={{Id}} selected="{{#if equal Id
../phase}}true{{/if}}">{{name}}</option>
      {{/each}}
    </select>
  </div>
  <h3>Checks</h3>
  {{#each groups}}
    <h4>{{_id}}. {{name}}
      <button class="btn btn-success" id={{_id}}>
        <span class="glyphicon glyphicon-plus"></span>
        New {{name}} Check
      </button>
    </h4>
    {{#each checks _id}}
      <div class="checkbox">
        <label><input type="checkbox" value="{{_id}}" checked="{{#if checked _id
../../checks}}true{{/if}}">{{_id}} - {{desc}}</label>
      </div>
    {{/each}}
  {{/each}}
  <br>
  <button type="button" id="save" class="btn btn-success btn-lg">
    <span class="glyphicon glyphicon-saved"></span>
    Save
  </button>
  <button type="button" id="cancel" class="btn btn-link">Cancel</button>
</template>

<template name="checklistInfo">
  <h1>Checklist</h1>
  <strong>Tollgate: </strong>{{title phase}}
  {{#if svn}}
    <br>
    <strong>SVN: </strong><a href={{svn}} target="_blank">{{svn}}</a>
  {{/if}}
  <hr>
  <div class="form-group">
    <label for="assistants">Assistants</label>
    <input class="form-control" type="text" id="assistants" disabled={{notAdmin}}
value={{assistants}}>
  </div>
  {{#each types}}
      <h2>{{groupName this}}</h2>
```

```
      {{> checklistTable data=.. tableType=this}}
  {{/each}}
  <div class="form-group">
    <label for="conclusions">Conclusions</label>
    <textarea class="form-control" rows="5" id="conclusions"
disabled={{notAdmin}}>{{conclusions}}</textarea>
  </div>
  <button type="button" id="save" class="btn btn-success btn-lg">
    <span class="glyphicon glyphicon-saved"></span>
    Save All
  </button>
</template>

<template name="checklistTable">
  <table class="table">
    <thead>
      <tr>
        <th class="col-xs-1">Id</th>
        <th class="col-xs-3 col-md-4">Description</th>
        <th class="col-xs-1">Applies</th>
        <th class="col-xs-2 col-md-1">Result</th>
        <th class="col-xs-3 col-md-3">Comments</th>
        <th class="col-xs-2 col-md-2">Criticity</th>
      </tr>
    </thead>
    <tbody>
      {{#each findChecks tableType }}
        {{> checkRow}}
      {{/each}}
    </tbody>
  </table>
  {{#if currentUser}}
    <button type="button" id="save" class="btn btn-success">
      <span class="glyphicon glyphicon-saved"></span>
      Save {{groupName tableType}}
    </button>
  {{/if}}
</template>
```

client/views/checks.html

```
<template name="checksList">
  <h1>Checks</h1>
  <button class="btn btn-success btn-lg" id="new">
    <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> New Group
  </button>
  {{#each groups}}
    {{> groupChecks}}
  {{/each}}
</template>
```

```
<template name="groupChecks">
  <h2>{{_id}}. {{name}}
    <button class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete Group
    </button>
  </h2>
  <button class="btn btn-success" id={{_id}}>
    <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> New Check
  </button>
  {{#each checks _id}}
    {{> checksRow}}
  {{/each}}
</template>

<template name="checksRow">
  <div class="row">
    <a href="/checks/{{_id}}">
      <div class="col-xs-4">
        <h4>{{_id}}</h4>
      </div>
      <div class="col-xs-8">
        <h4>{{desc}}</h4>
      </div>
    </a>
  </div>
</template>

<template name="checksInfo">
  <h1>{{_id}}
    <span>
      <a href="/checks/{{_id}}/edit" class="btn btn-default">
        <span class="glyphicon glyphicon-pencil" aria-hidden="true"></span> Edit
      </a>
    </span>
  </h1>
  <span class="form-control input-lg no-border">{{desc}}</span>
  <br>
  <br>
  <button type="button" class="btn btn-danger" id="delete">
    <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
  </button>
  <button type="button" class="btn btn-link" id="cancel">Back</button>
</template>

<template name="checksEdit">
  <h1>{{_id}}</h1>
  <input type="text" id="desc" class="form-control input-lg" value={{desc}}>
  <br>
  <button type="button" id="save" class="btn btn-success btn-lg">
    <span class="glyphicon glyphicon-saved"></span>
```

```
    Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>




<template name="checkRow">
<tr bgcolor={{color}}>
  <td>
    {{_id}}
  </td>
  <td>
    {{desc}}
  </td>
  <td>
    <input style="transform:scale(1.5)" type="checkbox" id="checkbox"
checked={{applicable}} disabled={{#unless currentUser}}true{{/unless}}>
  </td>
  <td>
    {{#unless currentUser}}
      <select class="form-control" disabled>
        <option value={{result}}>{{result}}</option>
      </select>
    {{else}}
      {{#if applicable}}
        <select class="form-control" id="result">
          <option value="" selected={{#if equal "" result}}true{{/if}}></option>
          <option value="OK" selected={{#if equal "OK" result}}true{{/if}}>OK</option>
          <option value="POK" selected={{#if equal "POK" result}}true{{/if}}>POK</option>
          <option value="NOK" selected={{#if equal "NOK" result}}true{{/if}}>NOK</option>
        </select>
      {{else}}
        <select class="form-control" disabled>
          <option value=""></option>
        </select>
      {{/if}}
    {{/unless}}
  </td>
  <td>
    <textarea class="form-control" rows="4" id={{_id}} disabled={{#unless
currentUser}}true{{/unless}}>{{comments}}</textarea>
  </td>
  <td>
    {{#unless currentUser}}
      <select class="form-control" disabled>
        <option value={{criticality}}>{{criticality}}</option>
      </select>
    {{else}}
      {{#if equal "OK" result}}
```

```
        <select class="form-control" disabled>
          <option value=""></option>
        </select>
      {{else}}
        {{#if equal "" result}}
          <select class="form-control" disabled>
            <option value=""></option>
          </select>
        {{else}}
          <select class="form-control" id="critical">
            <option value="" selected={{#if equal "" criticality}}true{{/if}}></option>
            <option value="Major" selected={{#if equal "Major"
criticality}}true{{/if}}>Major</option>
            <option value="Minor" selected={{#if equal "Minor"
criticality}}true{{/if}}>Minor</option>
          </select>
        {{/if}}
      {{/if}}
    {{/unless}}
  </td>
</tr>
</template>
```

client/views/events.html

```
<template name="newEvent">
  <h2>New Event</h2>
  <strong>Project: </strong>{{name}} ({{_id}})
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label for="name">Name <span style="color:red">*</span></label>
    <input type="text" class="form-control" id="name">
  </div>
  <div class="form-group">
    <label for="date">Date <span style="color:red">*</span></label>
    <input type="text" class="form-control" id="date">
  </div>
  <div class="row">
    <div class="col-md-6">
      <div class="form-group">
        <label for="type">Type</label>
        <select class="form-control" id="type">
          {{#each types2}}
            <option value="{{_id}}">{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
```

```
    <div class="col-md-6">
      <div class="form-group">
        <label for="category">Category</label>
        <select class="form-control" id="category">
          {{#each categories}}
            <option value="{{_id}}">{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="col-md-6">
      <div class="form-group">
        <label for="phase">Phase</label>
        <select class="form-control" id="phase">
          <option value=""></option>
          {{#each phases}}
            <option value="{{_id}}">{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
    <div class="col-md-6">
      <div class="form-group">
        <label for="release">Release</label>
        <select class="form-control" id="release">
          <option value=""></option>
          {{#each releases}}
            <option value="{{_id}}">{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
  </div>
  <div class="form-group">
    <label for="result">Result</label>
    <select class="form-control" id="result">
      <option value="" selected={{#if equal "" result}}true{{/if}}></option>
      <option value="OK" selected={{#if equal "OK" result}}true{{/if}}>OK</option>
      <option value="POK" selected={{#if equal "POK" result}}true{{/if}}>POK</option>
      <option value="NOK" selected={{#if equal "NOK" result}}true{{/if}}>NOK</option>
    </select>
  </div>

  <span style="color:red">*Required</span>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
```

```
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>


<template name="eventInfo">
  <h1>Event {{name}}
    {{#if currentUser.profile.admin}}
      <a href="/project/{{project}}/timeline/{{_id}}/edit" class="btn btn-default">
        <span class="glyphicon glyphicon-pencil" aria-hidden="true"></span> Edit
      </a>
    {{/if}}
  </h1>
  <strong>Project: </strong>{{project}}
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label>Date</label>
    <span class="form-control no-border">{{date}}</span>
  </div>
  <div class="row">
    <div class="col-md-6">
      <div class="form-group">
        <label>Type</label>
        <span class="form-control no-border">{{typeName type}}</span>
      </div>
    </div>
    <div class="col-md-6">
      <div class="form-group">
        <label>Category</label>
        <span class="form-control no-border">{{categoryName category}}</span>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="col-md-6">
      <div class="form-group">
        <label>Phase</label>
        <span class="form-control no-border">{{phaseName phase}}</span>
      </div>
     </div>
   <div class="col-md-6">
      <div class="form-group">
        <label>Release</label>
        <span class="form-control no-border">{{releaseName release}}</span>
      </div>
    </div>
  </div>
  <div class="form-group">
    <label for="result">Result</label>
    <span class="form-control no-border" style={{#if equal result
```

```
"OK"}}"color:green;"{{else}}{{#if equal result
"POK"}}"color:orange;"{{else}}"color:red;"{{/if}}{{/if}}>{{result}}</span>
  </div>
</template>



<template name="eventEdit">
  <h1>Event {{name}}
    <button type="button" class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
    </button>
  </h1>
  <strong>Project: </strong>{{project}}
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label for="name">Name</label>
    <input type="text" class="form-control" id="name" value={{name}}>
  </div>
  <div class="form-group">
    <label for="date">Date <span style="color:red">*</span></label>
    <input type="text" class="form-control" id="date" value={{date}}>
  </div>
  <div class="row">
    <div class="col-md-6">
      <div class="form-group">
        <label for="type">Type</label>
        <select class="form-control" id="type">
          {{#each types2}}
            <option value="{{_id}}" selected={{#if equal _id
../type}}true{{/if}}>{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
    <div class="col-md-6">
      <div class="form-group">
        <label for="category">Category</label>
        <select class="form-control" id="category">
          {{#each categories}}
            <option value="{{_id}}" selected={{#if equal _id
../category}}true{{/if}}>{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="col-md-6">
```

```
      <div class="form-group">
        <label for="phase">Phase</label>
        <select class="form-control" id="phase">
          <option value=""></option>
          {{#each phases}}
            <option value="{{_id}}" selected={{#if equal _id
../phase}}true{{/if}}>{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
    <div class="col-md-6">
      <div class="form-group">
        <label for="release">Release</label>
        <select class="form-control" id="release">
          <option value=""></option>
          {{#each releases}}
            <option value="{{_id}}" selected={{#if equal _id
../release}}true{{/if}}>{{name}}</option>
          {{/each}}
        </select>
      </div>
    </div>
  </div>
  <div class="form-group">
    <label for="result">Result</label>
    <select class="form-control" id="result">
      <option value="" selected={{#if equal "" result}}true{{/if}}></option>
      <option value="OK" selected={{#if equal "OK" result}}true{{/if}}>OK</option>
      <option value="POK" selected={{#if equal "POK" result}}true{{/if}}>POK</option>
      <option value="NOK" selected={{#if equal "NOK" result}}true{{/if}}>NOK</option>
    </select>
  </div>

  <span style="color:red">*Required</span>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>
```

client/views/home.html

```
<template name="home">
  <div class="row">
    {{#if currentUser}}
      <div class="col-xs-12">
        {{#if currentUser.profile.admin}}
```

```
          <a href="/project/new" class="btn btn-success">
            <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Create New
Project
          </a>
        {{/if}}
        <br>
        <br>
      </div>
    {{/if}}
    <div class="col-xs-2">
      {{#if closed}}
        <a id="switch-to-run" class="btn btn-link">Running Projects</a>
      {{else}}
        <a id="switch-to-close" class="btn btn-link">Closed Projects</a>
      {{/if}}
    </div>
    <div class="col-xs-7">
      <button style="opacity:1; cursor:default;" class="btn btn-success"
disabled>Passed</button>
      <button style="opacity:1; cursor:default;" class="btn btn-danger" disabled>Not
Passed</button>
      <button style="opacity:1; cursor:default;" class="btn btn-info"
disabled>Scheduled</button>
      <button style="opacity:1; cursor:default;" class="btn btn-warning" disabled>Over
Date</button>
    </div>
    <div class="col-xs-3">
      {{> drop}}
    </div>
  </div>
  <br>
  {{> projectTable projectPage=false}}
</template>

<template name="drop">
  <label for="dropdownMenu1">Year: </label>
  <div class="dropdown" style="display:inline">
    <button class="btn btn-default dropdown-toggle" type="button" id="dropdownMenu1"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="true">
      {{currentYear}}
      <span class="caret"></span>
    </button>
    <ul class="dropdown-menu" aria-labelledby="dropdownMenu1">
      {{#each years}}
        <li><a href="#">{{year}}</a></li>
      {{/each}}
    </ul>
  </div>
</template>
```

client/views/index.html

```
<head>
  <title>Idneo Quality Engine</title>
  <link rel="icon" type="image/png" href="/images/logo_idneo_favicon.png"/>
  <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">

  <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
</head>
```

client/views/KPIs.html

```
<template name="newKPI">
  <h2>New KPI</h2>
  <strong>Project: </strong>{{name}} ({{_id}})
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label for="date">Date</label>
    <input type="text" class="form-control" id="date">
  </div>
  {{#each targets}}
    <div class="form-group">
      <label for={{_id}}>{{name}}</label>
      <input type="number" class="form-control" id={{_id}}>
    </div>
  {{/each}}
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>


<template name="editKPI">
  <h2>Edit KPI
    <button class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete KPI
    </button>
  </h2>
  <strong>Project: </strong>{{project}}
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label for="date">Date</label>
    <input type="text" class="form-control" id="date" value={{date}}>
```

```
  </div>
  {{#each targets}}
    <div class="form-group">
      <label for={{_id}}>{{name}}</label>
      <input type="number" class="form-control" id={{_id}} value={{getTargetData ..}}>
    </div>
  {{/each}}
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>



<template name="KPIs">
  {{#if closed}}
    <a id="switch-to-run" class="btn btn-link">Running Projects</a>
  {{else}}
    <a id="switch-to-close" class="btn btn-link">Closed Projects</a>
  {{/if}}
  {{#each projects}}
    <h2><a href="/project/{{_id}}"><strong>{{name}}</strong></a>
      {{#if currentUser.profile.admin}}
        <a href="/project/{{_id}}/KPIs/new" class="btn btn-success">
          <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Add KPI
        </a>
      {{/if}}
    </h2>
    {{> lastKPI}}
    {{#with _id}}
      {{> chart}}
    {{/with}}
  {{/each}}
</template>

<template name="lastKPI">
  {{#if kpi}}
    <h4>Last sample (<em>week {{kpi.week}}</em>):</h4>
    {{#with kpi}}
      {{#each targets}}
        <strong>{{name}}</strong>: {{getTargetData ..}}
        <br>
      {{/each}}
    {{/with}}
  {{else}}
    <h4>No data</h4>
  {{/if}}
</template>
```

```
<template name="KPIsTable">
  <div class="outer-container">
        <div class="inner-container">
            <div class="table-header">
                <table id="headertable" class="table table-striped">
                  <thead>
                    <tr>
                      <th class="header-cell col1">
                        Date
                      </th>
                      {{#each targets}}
                        <th class="header-cell col1">{{name}}</th>
                      {{/each}}
                    </tr>
                  </thead>
                </table>
            </div>
            <div class="table-body" style="height: 166px;">
                <table id="bodytable" class="table table-striped">

                  <tbody>
                    {{#each kpis _id}}
                      <tr>
                          <td class="body-cell col1">
                            {{#if currentUser.profile.admin}}
                              <a href="/project/{{project}}/KPIs/{{_id}}">Week {{week}}
({{year}})</a>
                            {{else}}
                              Week {{week}} ({{year}})
                            {{/if}}
                          </td>
                          {{#each targets}}
                            <td class="body-cell col1">{{getTargetData ..}}</td>
                          {{/each}}
                      </tr>
                    {{/each}}
                  </tbody>
                </table>
            </div>
        </div>
    </div>

    <script type="text/javascript">
      $(document).ready(function () {
        $(".table-body").scroll(function ()
        {
          $(".table-header").offset({ left: -1*this.scrollLeft+$(this).offset().left});
        });
      });
```

```
  </script>
</template>

<template name="targets">
  <h1>Targets
    <button class="btn btn-success" id="new">
      <span class="glyphicon glyphicon-plus" aria-hidden="true" ></span> Create New
Target
    </button>
  </h1>
  <br>
  {{#each targets}}
    {{> targetInfo}}
  {{/each}}
</template>

<template name="targetInfo">
  <div class="row">
    <div class="col-xs-3">
      <h4>{{name}}</h4>
    </div>
    <div class="col-xs-2">
      <button type="button" class="btn btn-default" id="edit">Edit</button>
    </div>
    <div class="col-xs-2">
      <button type="button" class="btn btn-danger" id="delete">Delete</button>
    </div>
  </div>
</template>
```

client/views/layout.html

```
<template name="layout">
  <script type="text/javascript">
    google.charts.load('current', {'packages':['line']});
  </script>
  <div class="jumbotron">
    <div class="row">
      <div class="col-xs-3">
        <a href="/" class="align-left"><img src="/images/logo_idneo.png"
style="width:90%;"/></a>
      </div>
      <div class="col-xs-9">
        <h1>Quality Engine</h1>
      </div>
    </div>
  </div>
  <div class="container">
    {{#if logginIn}}
      Loading ...
```

```
    {{else}}
      <div class="pull-right">
        {{> navbar}}
      </div>
      <br>
      {{> yield}}
    {{/if}}
  </div>
</template>

<template name="navbar">
  <nav class="navbar navbar-default navbar-fixed-top">
    <div class="container">
      <div id="navbar" class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
          <li><a class="navbar-brand" href="/">Tollgate Evaluation</a></li>
          <li><a class="navbar-brand" href="/offers">Offers</a></li>
          <li><a class="navbar-brand" href="/KPIs">KPI's</a></li>
          {{#if currentUser}}
            <li class="dropdown">
              <a class="dropdown-toggle" data-toggle="dropdown" role="button" aria-
haspopup="true" aria-expanded="false">Configuration<span class="caret"></span></a>
              <ul class="dropdown-menu">
                <li><a href="/users">Users</a></li>
                {{#if currentUser.profile.admin}}
                  <li role="separator" class="divider"></li>
                  <li class="dropdown-header">Checklists</li>
                  <li><a href="/checklist/schemes">Schemes</a></li>
                  <li><a href="/checklist/templates">Templates</a></li>
                  <li><a href="/checks">Checks</a></li>
                  <li role="separator" class="divider"></li>
                  <li><a href="/KPIs/targets">KPIs Targets</a></li>
                  <li role="separator" class="divider"></li>
                  <li class="dropdown-header">Timeline</li>
                  <li><a href="/timeline/categories">Timeline Categories</a></li>
                  <li><a href="/events/types">Event Types</a></li>
                {{/if}}
              </ul>
            </li>
          {{/if}}
        </ul>
        <ul class="nav navbar-nav navbar-right">
          {{#if currentUser}}
            <li><button class="btn btn-primary"
id="link">{{currentUser.profile.name}}</button></li>
            <li><button type="button" class="btn btn-danger"
id="logout">Logout</button></li>
          {{else}}
            <li><button id="login" class="btn btn-success pull-right">Login</button></li>
          {{/if}}
        </ul>
```

```
      </div><!--/.nav-collapse -->
    </div>
  </nav>
</template>
```

client/views/offers.html

```
<template name="offers">
  {{#if currentUser.profile.admin}}
    <a href="/offers/new" class="btn btn-success">
      <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Create New Offer
    </a>
  {{/if}}
  {{#if showArchived}}
    <button type="button" id="unarchived" class="btn btn-link">Show unarchived
offers</button>
  {{else}}
    <button type="button" id="archived" class="btn btn-link">Show archived
offers</button>
  {{/if}}
  <table class="table">
    <thead>
      <tr>
        <th>Customer</th>
        <th>BU</th>
        <th>Product</th>
        <th>Scope</th>
        <th>Status</th>
        <th>Comments</th>
      </tr>
    </thead>
    <tbody>
      {{#each offers}}
        {{> offerRow}}
      {{/each}}
    </tbody>
  </table>
</template>

<template name="newOffer">
  <h1>New Offer</h1>
  <div class="form-group">
    <label for="cust">Customer <span style="color:red">*</span></label>
    <input class="form-control" type="text" id="cust" placeholder="Customer">
  </div>
  <div class="form-group">
    <label for="bu">BU</label>
    <input class="form-control" type="text" id="bu" placeholder="BU">
  </div>
  <div class="form-group">
```

```
    <label for="prod">Product</label>
    <input class="form-control" type="text" id="prod" placeholder="Product">
  </div>
  <div class="form-group">
    <label for="scope">Scope</label>
    <input class="form-control" type="text" id="scope" placeholder="Scope">
  </div>
  <div class="form-group">
    <label for="status">Status</label>
    <select class="form-control" id="status">
      {{> selectStatus}}
    </select>
  </div>
  <div class="form-group">
    <label for="svn">Link</label>
    <input type="text" class="form-control" id="svn" placeholder="Link to SVN">
  </div>
  <div class="form-group">
    <label for="comment">Comments</label>
    <textarea class="form-control" rows="4" id="comment"></textarea>
  </div>
  <span style="color:red">*Required</span>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <a href="/offers" class="btn btn-link">Cancel</a>
</template>


<template name="offerInfo">
  <h1>Edit Offer
    <button type="button" class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash"></span> Delete
    </button>
  </h1>
  <div class="form-group">
    <label for="cust">Customer</label>
    <input class="form-control" type="text" id="cust" placeholder="Customer"
value="{{customer}}">
  </div>
  <div class="form-group">
    <label for="bu">BU</label>
    <input class="form-control" type="text" id="bu" placeholder="BU" value="{{bu}}">
  </div>
  <div class="form-group">
    <label for="prod">Product</label>
    <input class="form-control" type="text" id="prod" placeholder="Product"
value="{{product}}">
  </div>
```

```
  <div class="form-group">
    <label for="scope">Scope</label>
    <input class="form-control" type="text" id="scope" placeholder="Scope"
value="{{scope}}">
  </div>
  <div class="form-group">
    <label for="status">Status</label>
    <select class="form-control" id="status">
      {{> selectStatus}}
    </select>
  </div>
  <div class="form-group">
    <label for="svn">Link</label>
    <input type="text" class="form-control" id="svn" placeholder="Link to SVN"
value={{svn}}>
  </div>
  <div class="form-group">
    <label for="comment">Comments</label>
    <textarea class="form-control" rows="4" id="comment">{{comments}}</textarea>
  </div>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <a href="/offers" class="btn btn-link">Cancel</a>
</template>




<template name="selectStatus">
  {{#each status}}
    <option value="{{name}}" selected={{#if equal name
../status}}true{{/if}}>{{name}}</option>
  {{/each}}
</template>




<template name="offerRow">
  <tr bgcolor={{color}}>
    <td>
      {{#if currentUser.profile.admin}}
        <a class="form-control" href="/offers/{{_id}}">
          <strong>{{customer}}</strong>
        </a>
```

```
        <br>
        <button type="button" class="btn btn-link" id="switch">{{#if
archived}}Unarchive{{else}}Archive{{/if}}</button>
      {{else}}
        <a class="form-control" href={{svn}}>
          <strong>{{customer}}</strong>
        </a>
      {{/if}}
    </td>
    <td>
      {{bu}}
    </td>
    <td>
      {{product}}
    </td>
    <td>
      {{scope}}
    </td>
    <td>
      <select class="form-control" id="status" disabled={{#unless
currentUser.profile.admin}}true{{/unless}}>
        {{> selectStatus}}
      </select>
    </td>
    <td>
      <textarea class="form-control" rows="4" id="comment" disabled={{#unless
currentUser.profile.admin}}true{{/unless}}>{{comments}}</textarea>
    </td>
  </tr>
</template>
```

client/views/phases.html

```
<template name="newPhase">
  <h2>New Phase</h2>
  <strong>Project: </strong>{{name}} ({{_id}})
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label for="name">Name <span style="color:red">*</span></label>
    <input type="text" class="form-control" id="name">
  </div>
  <div class="form-group">
    <label for="start">Date Range <span style="color:red">*</span></label>
    <div class="input-daterange input-group" id="datepicker">
      <input type="text" class="input-sm form-control" id="start" />
      <span class="input-group-addon">to</span>
      <input type="text" class="input-sm form-control" id="end" />
    </div>
```

```
  </div>
  <span style="color:red">*Required</span>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>


<template name="phaseInfo">
  <h1>Phase {{name}}
    {{#if currentUser.profile.admin}}
      <a href="/project/{{project}}/timeline/{{_id}}/edit" class="btn btn-default">
        <span class="glyphicon glyphicon-pencil" aria-hidden="true"></span> Edit
      </a>
    {{/if}}
  </h1>
  <strong>Project: </strong>{{project}}
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label>Date Range</label>
    <br>
    <span class="form-control no-border">{{start}} <strong>to</strong> {{end}}</span>
  </div>
  <div class="form-group">
    <label>State</label>
    <br>
    <span class="form-control no-border">
      {{#if open}}
        <span style="color:green;">Open</span>
      {{else}}
        <span style="color:red;">Close</span>
      {{/if}}
    </span>
  </div>
</template>

<template name="phaseEdit">
  <h1>Phase {{name}}
    <button type="button" class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
    </button>
  </h1>
  <hr>
  <br>
  <div class="form-group">
    <label for="name">Name</label>
```

```
    <input type="text" class="form-control" id="name" value={{name}}>
  </div>
  <div class="form-group">
    <label for="start">Date Range</label>
    <div class="input-daterange input-group" id="datepicker">
      <input type="text" class="input-sm form-control" id="start" value={{start}}>
      <span class="input-group-addon">to</span>
      <input type="text" class="input-sm form-control" id="end" value={{end}}>
    </div>
  </div>
  <button type="button" class="btn btn-primary" id="switch">{{#if open}}Close
Phase{{else}}Open Phase{{/if}}</button>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>
```

client/views/projects.html

```
<template name="createProject">
  <h1>New Project</h1>
  <div class="form-group">
    <label for="name">Project Name<span style="color:red">*</span></label>
    <input class="form-control" type="text" id="name" placeholder="Project Name">
  </div>
  <div class="form-group">
    <label for="ref">Reference Code<span style="color:red">*</span></label>
    <input class="form-control" type="text" id="ref" placeholder="Reference Code">
  </div>
  <div class="form-group">
    <label for="projectLead">Project Manager</label>
    <select class="form-control" id="projectLead">
      {{> selectManager}}
    </select>
  </div>
  <div class="row">
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="swLead">Software Lead</label>
        <select class="form-control" id="swLead">
          {{> selectUser}}
        </select>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="hwLead">Hardware Lead</label>
```

```
        <select class="form-control" id="hwLead">
          {{> selectUser}}
        </select>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="mechLead">Mechanical Lead</label>
        <select class="form-control" id="mechLead">
          {{> selectUser}}
        </select>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="system">System Architect</label>
        <select class="form-control" id="system">
          {{> selectUser}}
        </select>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="qa">QA Engineer</label>
        <select class="form-control" id="qa">
          {{> selectUser}}
        </select>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="tv">T&V Lead</label>
        <select class="form-control" id="tv">
          {{> selectUser}}
        </select>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="tr">T&R Lead</label>
        <select class="form-control" id="tr">
          {{> selectUser}}
        </select>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="compliance">Compliance Lead</label>
        <select class="form-control" id="compliance">
```

```
          {{> selectUser}}
        </select>
      </div>
    </div>
  </div>
  <div class="form-group">
    <label for="projectLead">Checklist Scheme</label>
    <select class="form-control" id="scheme">
      {{> selectScheme}}
    </select>
  </div>
  <span style="color:red">*Required</span>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="add">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <a href="/" class="btn btn-link">Cancel</a>
</template>


<template name="projectPage">
  <div class="btn-group" role="group" aria-label="...">
    <a class="btn btn-primary" href="/project/{{_id}}">General</a>
    <a class="btn btn-default" href="/project/{{_id}}/timeline">Timeline</a>
  </div>
  {{> projectInfo}}
  {{#if currentUser.profile.admin}}
    <button class="btn btn-success" id="add-release">
      <span class="glyphicon glyphicon-plus" aria-hidden="true"></span>
      Add Release
    </button>
  {{/if}}
  <br>
  {{> projectTable projectPage=true}}
  <br>
  <hr>
  <br>
  <h3>KPIs
    {{#if currentUser.profile.admin}}
      <a href="/project/{{_id}}/KPIs/new" class="btn btn-success">
        <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Add KPI
      </a>
    {{/if}}
  </h3>
  {{> KPIsTable}}
  <hr>
  <h3>KPI Line Chart</h3>
  <br>
  {{#with _id}}
    {{> chart}}
```

```
  {{/with}}
</template>



<template name="projectInfo">
  <h1>{{name}}
    {{#if currentUser.profile.admin}}
      <a href="/project/{{_id}}/edit" class="btn btn-default">
        <span class="glyphicon glyphicon-pencil" aria-hidden="true"></span> Edit
      </a>
    {{/if}}
  </h1>
  {{_id}}
  <br>
  <br>
  <div class="form-group">
    <label for="projectLead">Project Manager</label>
    <span class="form-control no-border">{{fullName projectLead}}</span>
  </div>
  <div class="row">
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="projectLead">Software Lead</label>
        <span class="form-control no-border">{{fullName swLead}}</span>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="projectLead">Hardware Lead</label>
        <span class="form-control no-border">{{fullName hwLead}}</span>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="projectLead">Mechanical Lead</label>
        <span class="form-control no-border">{{fullName mechLead}}</span>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="projectLead">System Architect</label>
        <span class="form-control no-border">{{fullName system}}</span>
      </div>
    </div>
  </div>
  <div class="row">
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="qa">QA Engineer</label>
        <span class="form-control no-border">{{fullName QA}}</span>
```

```
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="tv">T&V Lead</label>
        <span class="form-control no-border">{{fullName TV}}</span>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="tr">T&R Lead</label>
        <span class="form-control no-border">{{fullName TR}}</span>
      </div>
    </div>
    <div class="col-xs-6 col-md-3">
      <div class="form-group">
        <label for="compliance">Compliance Lead</label>
        <span class="form-control no-border">{{fullName compliance}}</span>
      </div>
    </div>
  </div>
  <div class="form-group">
    <label for="projectLead">Checklist Scheme</label>
    <span class="form-control no-border">{{schemeName scheme}}</span>
  </div>
</template>









<template name="projectTable">
  <table class="table table-bordered">
    <thead>
      <tr>
        {{#unless projectPage}}
        <th rowspan="2">
          {{#if tollgates}}
            <a class="btn btn-link" id="show-toll">Show tollgates</a>
          {{else}}
            <a class="btn btn-link" id="hide-toll">Hide tollgates</a>
          {{/if}}
          <br>
          {{#if showAll}}
            <a class="btn btn-link" id="show-all">Show all projects</a>
          {{else}}
            <a class="btn btn-link" id="hide-all">Hide all projects</a>
          {{/if}}
```

```
          <br>
          Project Info
        </th>
        {{/unless}}
        <!--<th rowspan="2">Release</th>-->
        <th colspan="2">Phase 3<br>(Alpha)</th>
        <th colspan="2">Phase 4<br>(Beta)</th>
        <th colspan="2">Phase 5<br>(Pre-Production)</th>
        <th colspan="2">Phase 6<br>(Production)</th>
        {{#if projectPage}}
          {{#if currentUser.profile.admin}}
            <th rowspan="2">Delete</th>
          {{/if}}
        {{/if}}
      </tr>
      <tr>
        <th>Frontload</th>
        <th>Closure</th>
        <th>Frontload</th>
        <th>Closure</th>
        <th>Frontload</th>
        <th>Closure</th>
        <th>Frontload</th>
        <th>Closure</th>
      </tr>
    </thead>
    <tbody>
      {{#if projectPage}}
        {{> projectRow ..}}
      {{else}}
        {{#each projects}}
          {{> projectRow}}
        {{/each}}
      {{/if}}
    </tbody>
  </table>
</template>


<template name="projectEdit">
    <h1>{{name}}
      <button type="button" class="btn btn-danger" id="delete">
        <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
      </button>
    </h1>
    {{_id}}
    <br>
    <br>
    <div class="form-group">
      <label for="projectLead">Project Manager</label>
      <select class="form-control" id="projectLead">
```

```
        {{> selectManager field="projectLead"}}
      </select>
    </div>
    <div class="row">
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="swLead">Software Lead</label>
          <select class="form-control" id="swLead">
            {{> selectUser field="swLead"}}
          </select>
        </div>
      </div>
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="hwLead">Hardware Lead</label>
          <select class="form-control" id="hwLead">
            {{> selectUser field="hwLead"}}
          </select>
        </div>
      </div>
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="mechLead">Mechanical Lead</label>
          <select class="form-control" id="mechLead">
            {{> selectUser field="mechLead"}}
          </select>
        </div>
      </div>
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="system">System Architect</label>
          <select class="form-control" id="system">
            {{> selectUser field="system"}}
          </select>
        </div>
      </div>
    </div>
    <div class="row">
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="qa">QA Engineer</label>
          <select class="form-control" id="qa">
            {{> selectUser field="QA"}}
          </select>
        </div>
      </div>
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="tv">T&V Lead</label>
          <select class="form-control" id="tv">
            {{> selectUser field="TV"}}
```

```
          </select>
        </div>
      </div>
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="tr">T&R Lead</label>
          <select class="form-control" id="tr">
            {{> selectUser field="TR"}}
          </select>
        </div>
      </div>
      <div class="col-xs-6 col-md-3">
        <div class="form-group">
          <label for="compliance">Compliance Lead</label>
          <select class="form-control" id="compliance">
            {{> selectUser field="compliance"}}
          </select>
        </div>
      </div>
    </div>
    <div class="form-group">
      <label for="scheme">Checklist Scheme</label>
      <select class="form-control" id="scheme">
        {{> selectScheme field="scheme"}}
      </select>
    </div>
    <button type="button" class="btn btn-success" id="save">
      <span class="glyphicon glyphicon-saved"></span>Save
    </button>
    {{#if archived}}
      <button type="button" class="btn btn-primary" id="open">Open Project</button>
    {{else}}
      <button type="button" class="btn btn-primary" id="close">Close Project</button>
    {{/if}}
    <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>









<template name="projectRow">
  {{#if isyear years createdAt}}
    {{#unless ../projectPage}}
      <tr>
        {{#if hide _id}}
          <td scope="row">
            <a href="/project/{{_id}}"><strong>{{name}}</strong>
```

```
            </a>
            <button type="button" class="btn btn-link" id="show">Show</button>
          </td>
        {{else}}
          <td scope="row" rowspan={{length}}>
            <a href="/project/{{_id}}"><strong>{{name}}</strong>
              <br>
              {{_id}}
              <br>
              <i>PM: {{fullName projectLead}}</i>
            </a>
            <button type="button" class="btn btn-link" id="hide">Hide</button>
          </td>
        {{/if}}
      </tr>
    {{/unless}}
    {{#each releases}}
      {{> releaseRow}}
    {{/each}}
  {{/if}}
</template>
```

client/views/releases.html

```
<template name="releaseRow">
  <tr>
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase3FL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase3Front ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase3CL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase3Close ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase4FL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase4Front ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
```

```
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase4CL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase4Close ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase5FL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase5Front ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase5CL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase5Close ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase6FL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase6Front ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
    <td>
      {{#if currentUser.profile.admin}}
        <a href="/{{_id}}/tollgate/new/phase6CL" class="btn btn-default">+</a>
      {{/if}}
      {{#each phase6Close ../../projectPage}}
        {{> tollgateCel}}
      {{/each}}
    </td>
    {{#if ../../projectPage}}
      {{#if currentUser.profile.admin}}
        <td>
          <button type="button" class="btn btn-danger" id="delete">
            <span class="glyphicon glyphicon-trash" aria-hidden="true"></span>
          </button>
        </td>
      {{/if}}
    {{/if}}
  </tr>
</template>
```

client/views/releaseTime.html

```
<template name="newReleaseTime">
  <h2>New Release</h2>
  <strong>Project: </strong>{{name}} ({{_id}})
  <br>
  <hr>
  <br>
  <div class="form-group">
    <label for="name">Name <span style="color:red">*</span></label>
    <input type="text" class="form-control" id="name">
  </div>
  <div class="form-group">
    <label for="start">Date Range <span style="color:red">*</span></label>
    <div class="input-daterange input-group" id="datepicker">
      <input type="text" class="input-sm form-control" id="start" />
      <span class="input-group-addon">to</span>
      <input type="text" class="input-sm form-control" id="end" />
    </div>
  </div>
  <div class="form-group">
    <label for="phase">Phase</label>
    <select class="form-control" id="phase">
      <option value=""></option>
      {{#each phases}}
        <option value="{{_id}}">{{name}}</option>
      {{/each}}
    </select>
  </div>
  <span style="color:red">*Required</span>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>


<template name="releaseTimeInfo">
  <h1>Release {{name}}
    {{#if currentUser.profile.admin}}
      <a href="/project/{{project}}/timeline/{{_id}}/edit" class="btn btn-default">
        <span class="glyphicon glyphicon-pencil" aria-hidden="true"></span> Edit
      </a>
    {{/if}}
  </h1>
  <strong>Project: </strong>{{project}}
  <br>
  <hr>
```

```
  <br>
  <div class="form-group">
    <label>Date Range</label>
    <br>
    {{start}} <strong>to</strong> {{end}}
  </div>
  <div class="form-group">
    <label>Phase</label>
    <span class="form-control no-border">{{phaseName phase}}</span>
  </div>
</template>

<template name="releaseTimeEdit">
  <h1>Release {{name}}
    <button type="button" class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
    </button>
  </h1>
  <hr>
  <br>
  <div class="form-group">
    <label for="name">Name</label>
    <input type="text" class="form-control" id="name" value={{name}}>
  </div>
  <div class="form-group">
    <label for="start">Date Range</label>
    <div class="input-daterange input-group" id="datepicker">
      <input type="text" class="input-sm form-control" id="start" value={{start}}>
      <span class="input-group-addon">to</span>
      <input type="text" class="input-sm form-control" id="end" value={{end}}>
    </div>
  </div>
  <div class="form-group">
    <label for="phase">Phase</label>
    <select class="form-control" id="phase">
      <option value=""></option>
      {{#each phases}}
        <option value="{{_id}}" selected={{#if equal _id
../phase}}true{{/if}}>{{name}}</option>
      {{/each}}
    </select>
  </div>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>
```

client/views/schemes.html

```
<template name="schemesList">
  <h1>Checklist Schemes</h1>
  <a href="/checklist/schemes/new" class="btn btn-success">
    <span class="glyphicon glyphicon-plus"></span>
    New Scheme
  </a>
  <br>
  <br>
  {{#each schemes}}
    <h3><a href="/checklist/schemes/{{_id}}">{{name}}</a></h3>
  {{/each}}
</template>

<template name="newScheme">
  <h1>New Checklist Scheme</h1>
  <div class="form-group">
    <label for="phase">Name</label>
    <input type="text" class="form-control" id="name" placeholder="Scheme Name">
  </div>
  <h3>Templates</h3>
  {{#each phases}}
    <h4>{{name}}</h4>
    <select class="form-control" id="{{Id}}">
      {{#each templates Id}}
        <option value={{_id}}>{{name}}</option>
      {{/each}}
    </select>
  {{/each}}
  <br>

  <button type="button" id="save" class="btn btn-success btn-lg">
    <span class="glyphicon glyphicon-saved"></span>
    Save
  </button>
  <button type="button" id="cancel" class="btn btn-link">Cancel</button>
</template>

<template name="schemeEdit">
  <h1>
    {{name}}
    <button type="button" class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
    </button>
  </h1>
  <div class="form-group">
    <label for="phase">Name</label>
    <input type="text" class="form-control" id="name" placeholder="Scheme Name"
value="{{name}}">
  </div>
```

```
  <h3>Templates</h3>
  {{#each phases}}
    <h4>{{name}}</h4>
    <select class="form-control" id="{{Id}}">
      {{#each templates Id}}
        <option value={{_id}} selected="{{#if equal ../../checklists ../Id
_id}}true{{/if}}">{{name}}</option>
      {{/each}}
    </select>
  {{/each}}
  <br>

  <button type="button" id="save" class="btn btn-success btn-lg">
    <span class="glyphicon glyphicon-saved"></span>
    Save
  </button>
  <button type="button" id="cancel" class="btn btn-link">Cancel</button>
</template>


<template name="selectScheme">
  {{#each schemes}}
    <option value="{{_id}}" selected={{select ../.. ../field}}>{{name}}</option>
  {{/each}}
</template>
```

client/views/timeline.html

```
<template name="timeline">
  <div class="btn-group" role="group" aria-label="...">
    <a class="btn btn-default" href="/project/{{_id}}">General</a>
    <a class="btn btn-primary" href="/project/{{_id}}/timeline">Timeline</a>
  </div>

  {{> projectInfo}}
  <h2>Timeline
    {{#if currentUser.profile.admin}}
      <a href="/project/{{_id}}/phases/new" class="btn btn-success">
        <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Add Phase
      </a>
      <a href="/project/{{_id}}/releases/new" class="btn btn-success">
        <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Add Release
      </a>
      <a href="/project/{{_id}}/events/new" class="btn btn-success">
        <span class="glyphicon glyphicon-plus" aria-hidden="true"></span> Add Event
      </a>
    {{/if}}
  </h2>
  {{#if Template.subscriptionsReady}}
    {{> timelineScheme}}
```

```
  {{else}}
    Loading...
  {{/if}}
</template>

<template name="timelineScheme">

  <div id="visualization"></div>
  <div style="height:200px"></div>
</template>

<template name="timelineInfo">
  {{#if equal type "Phase"}}
    {{#with data}}
      {{> phaseInfo}}
    {{/with}}
  {{else}}
    {{#if equal type "Release"}}
      {{#with data}}
        {{> releaseTimeInfo}}
      {{/with}}
    {{else}}
      {{#with data}}
        {{> eventInfo}}
      {{/with}}
    {{/if}}
  {{/if}}
</template>

<template name="timelineEdit">
  {{#if equal type "Phase"}}
    {{#with data}}
      {{> phaseEdit}}
    {{/with}}
  {{else}}
    {{#if equal type "Release"}}
      {{#with data}}
        {{> releaseTimeEdit}}
      {{/with}}
    {{else}}
      {{#with data}}
        {{> eventEdit}}
      {{/with}}
    {{/if}}
  {{/if}}
</template>
```

client/views/tollgates.html

```
<template name="newTollgate">
  <h2>{{title phase}}</h2>
  <strong>Project: </strong>{{project.name}}
  <br>
  <strong>Code: </strong>{{project._id}}
  <hr>
  <br>
  <div class="form-group">
    <label for="date">Date</label>
    <input type="text" class="form-control" id="date">
  </div>
  <div class="form-group">
    <label for="linkSVN">Link</label>
    <input type="text" class="form-control" id="linkSVN" placeholder="Link to SVN">
  </div>
  <div class="form-group">
    <label for="result">Result</label>
    <select id="result" class="form-control">
      {{#each results}}
        <option value={{name}}>{{name}}</option>
      {{/each}}
    </select>
  </div>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>

<template name="tollgateInfo">
  <h2>{{title phase}}
    <button type="button" class="btn btn-danger" id="delete">
      <span class="glyphicon glyphicon-trash" aria-hidden="true"></span> Delete
    </button>
  </h2>
  <strong>Project: </strong>{{project.name}}
  <br>
  <strong>Code: </strong>{{project._id}}
  <hr>
  <br>
  <div class="form-group">
    <label for="date">Date</label>
    <input type="text" class="form-control" id="date" value={{date}}>
  </div>
  <div class="form-group">
    <label for="linkSVN">Link</label>
    <input type="text" class="form-control" id="linkSVN" placeholder="Link to SVN"
value={{link}}>
```

```
  </div>
  <div class="form-group">
    <label for="result">Result</label>
    <select id="result" class="form-control">
      {{#each results}}
        <option value={{name}} selected="{{selected name ../result}}">{{name}}</option>
      {{/each}}
    </select>
  </div>
  {{#if checklist}}
    <a href="/checklist/{{checklist}}" type="button" class="btn btn-primary btn-lg">
      <span class="glyphicon glyphicon-list"></span>
      Checklist
    </a>
  {{/if}}
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">
    <span class="glyphicon glyphicon-saved"></span>Save
  </button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>

<template name="tollgateCel">
    {{#if currentUser.profile.admin}}
        <a href="/{{../_id}}/tollgate/{{_id}}" class={{color}}>{{empty}}</a>
    {{else}}
      {{#if checked}}
        <a href="/checklist/{{checklist}}" class={{color}}>{{empty}}</a>
      {{else}}
        {{#if linked}}
          <a href="{{link}}" target="_blank" class={{color}}>{{empty}}</a>
        {{else}}
          <a class={{color}}>{{empty}}</a>
        {{/if}}
      {{/if}}
    {{/if}}
    <br>
</template>
```

client/views/types.html

```
<template name="types">
  <h1>Types
    <button class="btn btn-success" id="new">
      <span class="glyphicon glyphicon-plus" aria-hidden="true" ></span> Create New Type
    </button>
  </h1>
  <br>
  {{#each types2}}
```

```
    {{> typeInfo}}
  {{/each}}
</template>

<template name="typeInfo">
  <div class="row">
    <div class="col-xs-3">
      <h4>{{name}}</h4>
    </div>
    <div class="col-xs-2">
      <button type="button" class="btn btn-default" id="edit">Edit</button>
    </div>
    <div class="col-xs-2">
      <button type="button" class="btn btn-danger" id="delete">Delete</button>
    </div>
  </div>
</template>
```

client/views/users.html

```
<template name="usersLogIn">
  <div class="form-group">
    <label for="username">Username</label>
    <input class="form-control" type="text" id="username" placeholder="Username">
  </div>
  <div class="form-group">
    <label for="password">Password</label>
    <input type="password" class="form-control" id="password" placeholder="Password">
  </div>
  <button type="button" class="btn btn-success" id="login">Login</button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>

<template name="createUser">
  <div class="form-group">
    <label for="username">Username</label>
    <input class="form-control" type="text" id="username" placeholder="Username">
  </div>
  <div class="form-group">
    <label for="password">Password</label>
    <input type="password" class="form-control" id="password" placeholder="Password">
  </div>
  <div class="form-group">
    <label for="password2">Repeat Password</label>
    <input type="password" class="form-control" id="password2" placeholder="Repeat
Password">
  </div>
  <div class="form-group">
    <label for="full-name">Full Name</label>
    <input class="form-control" type="text" id="full-name" placeholder="Full Name">
```

```
  </div>
  <div class="checkbox">
    <label>
      <input style="transform:scale(1.5)" id="manager" type="checkbox"> <strong>Project
Manager</strong>
    </label>
  </div>
  <div class="checkbox">
    <label>
      <input style="transform:scale(1.5)" id="admin" type="checkbox">
<strong>Admin</strong>
    </label>
  </div>
  <br>
  <button type="button" class="btn btn-success" id="add">Add</button>
  <a href="/users" class="btn btn-link">Cancel</a>
</template>

<template name="usersList">
  <div class="row">
    <div class="col-xs-12">
      {{#if currentUser.profile.admin}}
      <a href="/users/new" class="btn btn-success">
        <span class="glyphicon glyphicon-plus"></span>
        Create User
      </a>
      {{/if}}
    </div>
  </div>
  <br>
  <br>
  {{#if Template.subscriptionsReady}}
    {{> usersTable}}
  {{else}}
    Loading...
  {{/if}}
</template>

<template name="usersTable">
  <table class="table table-bordered">
    <thead>
      <tr>
        <th>Username</th>
        <th>Full Name</th>
        {{#if currentUser.profile.admin}}
          <th>Edit</th>
          <th>Delete</th>
        {{/if}}
      </tr>
    </thead>
    <tbody>
```

```
      {{#each users}}
        {{> userRow}}
      {{/each}}
    </tbody>
  </table>
</template>

<template name="userRow">
  <tr>
    <td>
      {{#if itsMe}}
        <a href="/users/{{currentUser._id}}">{{username}}</a>
      {{else}}
        {{username}}
      {{/if}}
    </td>
    <td>
      {{#if itsMe}}
        <a href="/users/{{currentUser._id}}">{{profile.name}}</a>
      {{else}}
        {{profile.name}}
      {{/if}}

    </td>
    {{#if currentUser.profile.admin}}
      <td><a href="/users/{{_id}}" class="btn btn-default"><span class="glyphicon
glyphicon-pencil" aria-hidden="true"></span></a></td>
      <td><a type="button" class="btn btn-danger" id="delete"><span class="glyphicon
glyphicon-trash" aria-hidden="true"></span></a></td>
    {{/if}}
  </tr>
</template>

<template name="userInfo">
  <div class="form-group">
    <label for="username">Username</label>
    <input class="form-control" type="text" id="username" placeholder="Username"
value="{{username}}" disabled="{{itsMe}}">
  </div>
  <div class="form-group">
    <label for="full-name">Full Name</label>
    <input class="form-control" type="text" id="full-name" placeholder="Full Name"
value="{{profile.name}}">
  </div>
  <div class="checkbox">
    <label>
      <input style="transform:scale(1.5)" id="manager" type="checkbox"
checked="{{profile.manager}}" disabled="{{itsMe}}"> <strong>Project Manager</strong>
    </label>
  </div>
  {{#if currentUser.profile.admin}}
```

```
    <div class="checkbox">
      <label>
        <input style="transform:scale(1.5)" id="admin" type="checkbox"
checked="{{profile.admin}}" disabled="{{itsMe}}"> <strong>Admin</strong>
      </label>
    </div>
  {{/if}}
  <button type="button" class="btn btn-primary" id="changePsw">Change Password</button>
  <br>
  <br>
  <button type="button" class="btn btn-success" id="save">Save</button>
  <button type="button" class="btn btn-link" id="cancel">Cancel</button>
</template>

<template name="selectUser">
  <option value=""></option>
  {{#each users}}
    <option value="{{_id}}" selected={{select ../.. ../field}}>{{profile.name}}</option>
  {{/each}}
</template>

<template name="selectManager">
  <option value=""></option>
  {{#each managers}}
    <option value="{{_id}}" selected={{select ../.. ../field}}>{{profile.name}}</option>
  {{/each}}
</template>
```

client/helpers/registerHelper.js

```
Template.registerHelper("types2", function(){
  return Types.find({},{sort:{name:1}});
});

Template.registerHelper("equal", function(a, b){
    return (a==b);
});

Template.registerHelper("categories", function(){
  return Categories.find({},{sort:{name:1}});
});

Template.registerHelper("targets", function(){
  return Targets.find({},{sort:{name:1}});
});

Template.registerHelper("notAdmin", function(){
  if(Meteor.user()){
    return(!Meteor.user().profile.admin);
```

```
  }else{
    return true;
  }
});

Template.registerHelper("groupName", function(Id){
  group=checksGroups.findOne({_id:Id});
  if(group){
    return group.name;
  }
});

Template.registerHelper("status", function(){
  var res = [
    {name:"Offer"},
    {name:"Awarded"},
    {name:"Dropped"}
  ];
  return res;
});

Template.registerHelper("results", function(){
  var res = [
    {name:"Scheduled"},
    {name:"Passed"},
    {name:"Not Passed"},
    {name:"Undefined"}
  ];
  return res;
});

Template.registerHelper("phases", function(){
  var res = [
    {name:"Phase 3 (Alpha) - Frontload", Id:"phase3FL"},
    {name:"Phase 3 (Alpha) - Closure", Id:"phase3CL"},
    {name:"Phase 4 (Beta) - Frontload", Id:"phase4FL"},
    {name:"Phase 4 (Beta) - Closure", Id:"phase4CL"},
    {name:"Phase 5 (Pre-Production) - Frontload", Id:"phase5FL"},
    {name:"Phase 5 (Pre-Production) - Closure", Id:"phase5CL"},
    {name:"Phase 6 (Production) - Frontload", Id:"phase6FL"},
    {name:"Phase 6 (Production) - Closure", Id:"phase6CL"}
  ];
  return res;
});

Template.registerHelper("title", function(Id){
  res = Blaze._globalHelpers.phases();
  for(i=0;i<res.length;i++){
    if(res[i].Id==Id){
      return res[i].name
```

```
    }
  }
});


Template.registerHelper("currentYear", function(){
  return Session.get("year");
});

Template.registerHelper("groups", function(){
  return checksGroups.find({}, {sort:{_id:1}});
});


Template.registerHelper("checks", function(type){
  //return Checks.find({type:type}, {sort:{num:1}});
  return Checks.find({type:type}, {sort:{_id:1}});
});

Template.registerHelper("users", function(){
  return Meteor.users.find({username:{$not:"admin"}}, {sort: {"profile.name":1}});
});

Template.registerHelper("isyear", function(array, timestamp){
  year=Session.get("year");
  return (array.indexOf(year)>-1 || year==timestamp)
});

Template.registerHelper("notManagers", function(){
  return Meteor.users.find({"profile.admin":false, "profile.manager":false}, {sort:
{"profile.name":1}});
});

Template.registerHelper("managers", function(){
  return Meteor.users.find({"profile.admin":false, "profile.manager":true}, {sort:
{"profile.name":1}});
});

Template.registerHelper("fullName", function(argument){
  user = Meteor.users.findOne({_id:argument});
  if (user){
    return user.profile.name;
  }else{
    return "";
  }
});

Template.registerHelper("hide", function(projectId){
  list = Session.get("hide");
```

```
  if (list){
    if (list.indexOf(projectId)>-1){
      return true;
    }
  }
  return false;
});
```

client/helpers/V1Helpers.js

```
Template.drop.helpers({
  years: function(){
    return Years.find({},{sort:{year:1}});
  }
});

Template.home.helpers({
  closed: function(){
    return Session.get("closedProjects");
  }
});

Template.projectTable.helpers({
  projects: function(){
    projects=Session.get("closedProjects");
    if (projects){
      return Projects.find({archived:true},{sort:{name:1}});
    }else{
      return Projects.find({archived:false},{sort:{name:1}});
    }
  },
  showAll:function(){
    val = Session.get("showAll");
    if(val=='true'){
      return true;
    }else{
      return false;
    }
  },
  tollgates:function(){
    val = Session.get("tollgates");
    if(val=='true'){
      return true;
    }else{
      return false;
    }
  }
});

Template.projectRow.helpers({
```

```
  releases: function(){
    if (!Template.parentData(1).projectPage){
      list=Session.get("hide");
      if (list.indexOf(this._id)>-1){
        return [];
      }else{
        projects=Session.get("closedProjects");
        if (projects){
          return Releases.find({project:this._id}).map(function(rel){return
rel}).reverse();
        }else{
          return Releases.find({project:this._id,
archived:false}).map(function(rel){return rel}).reverse();
        }
      }
    }else{
      return Releases.find({project:this._id}).map(function(rel){return rel}).reverse();
    }
  },
  length:function(){
    projects=Session.get("closedProjects");
    if (projects){
      return (Releases.find({project:this._id}).count()+1);
    }else{
      return (Releases.find({project:this._id, archived:false}).count()+1);
    }
  },
  fullName:function(id){
    user=Meteor.users.findOne({_id:id});
    if (user){
      return user.profile.name;
    }
  },
  years:function(){
    data=[];
    Releases.find({project:this._id, archived:false}).forEach(function(rel){
      data=data.concat(Tollgates.find({release:rel._id}).map(function(toll){return
toll.date.split("/")[2]}));
    });
    return data
  }
});

Template.releaseRow.helpers({
  phase3Front: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase3FL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
```

```
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase3FL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
          return [toll];
        }else{
          return [];
        }
      }else{
        return
Tollgates.find({release:this._id,phase:"phase3FL"}).map(function(toll){return
toll}).reverse();
      }
    }
  },
  phase3Close: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase3CL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase3CL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
          return [toll];
        }else{
          return [];
        }
      }else{
        return
Tollgates.find({release:this._id,phase:"phase3CL"}).map(function(toll){return
toll}).reverse();
      }
    }
  },
  phase4Front: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase4FL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase4FL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
```

```
          return [toll];
        }else{
          return [];
        }
      }else{
        return
Tollgates.find({release:this._id,phase:"phase4FL"}).map(function(toll){return
toll}).reverse();
      }
    }
  },
  phase4Close: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase4CL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase4CL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
          return [toll];
        }else{
          return [];
        }
      }else{
        return
Tollgates.find({release:this._id,phase:"phase4CL"}).map(function(toll){return
toll}).reverse();
      }
    }
  },
  phase5Front: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase5FL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase5FL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
          return [toll];
        }else{
          return [];
        }
      }else{
```

```
        return
Tollgates.find({release:this._id,phase:"phase5FL"}).map(function(toll){return
toll}).reverse();
      }
    }
  },
  phase5Close: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase5CL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase5CL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
          return [toll];
        }else{
          return [];
        }
      }else{
        return
Tollgates.find({release:this._id,phase:"phase5CL"}).map(function(toll){return
toll}).reverse();
      }
    }
  },
  phase6Front: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase6FL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase6FL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
          return [toll];
        }else{
          return [];
        }
      }else{
        return
Tollgates.find({release:this._id,phase:"phase6FL"}).map(function(toll){return
toll}).reverse();
      }
    }
```

```
  },
  phase6Close: function(projectPage){
    if (projectPage){
      return
Tollgates.find({release:this._id,phase:"phase6CL"}).map(function(toll){return
toll}).reverse();
    }else{
      hide=Session.get("tollgates");
      if(hide=="true"){
        toll =
Tollgates.find({release:this._id,phase:"phase6CL"}).map(function(toll){return
toll}).reverse()[0];
        if (toll){
          return [toll];
        }else{
          return [];
        }
      }else{
        return
Tollgates.find({release:this._id,phase:"phase6CL"}).map(function(toll){return
toll}).reverse();
      }
    }
  },
});

Template.tollgateCel.helpers({
  empty:function(){
    if(this.date==""){
      return "--/--/----";
    }else{
      return this.date;
    }
  },
  linked:function(){
    if(this.link==""){
      return false;
    }else{
      return true
    }
  },
  checked:function(){
    if(this.checklist){
      return true;
    }else{
      return false
    }
  },
  projectRef:function(){
    return Router.current().params.ref;
  },
```

```
  releaseId:function(){
    return Router.current().params.relId;
  },
  color:function(){
    if(this.result=="Passed"){
      return "btn btn-success"
      //return "#7FF083"; //green
    }else if (this.result=="Not Passed"){
      return "btn btn-danger"
      //return "#F07F8E"; //red
    }else if (this.result=="Pending"){
      if(isFuture(this.date)){
        return "btn btn-primary"
      }else{
        return "btn btn-warning"
      }  //return "#817FF0"; //blue
    }else if (this.result=="Scheduled"){
      if(isFuture(this.date)){
        return "btn btn-info"
      }else{
        return "btn btn-warning"
      }  //return "#7FF0E6"; //blue2
    }else{
      return "btn btn-default";
    }
  }
});

function isFuture(milestoneDate){
  date = new Date();
  year=milestoneDate.split('/')[2];
  month=milestoneDate.split('/')[1];
  day=milestoneDate.split('/')[0];
  if(date.getFullYear() > year){
    return false;
  }else if (date.getFullYear() == year){
    if (date.getMonth()+1 > month){
      return false;
    }else if (date.getMonth()+1 == month){
      if (date.getDate() > day){
        return false;
      }else{
        return true;
      }
    }else{
      return true;
    }
  }else{
    return true;
  }
};
```

```
Template.newTollgate.helpers({
  project: function(){
    release=Releases.findOne({_id:this.release});
    if(release){
      return Projects.findOne({_id:release.project});
    }
  },
  releaseName:function(){
    release = Releases.findOne({_id:this.release});
    if  (release){
      return release.name;
    }
  },
});

Template.tollgateInfo.helpers({
  empty:function(){
    if(this.date==""){
      return true;
    }else{
      return false
    }
  },
  project: function(){
    release=Releases.findOne({_id:this.release});
    if(release){
      return Projects.findOne({_id:release.project});
    }
  },
  releaseName:function(){
    release = Releases.findOne({_id:this.release});
    if  (release){
      return release.name;
    }
  },
  selected:function(a, b){
    if (a == b){
      return "selected"
    }
  }
});

Template.selectUser.helpers({
  select: function(project, field){
    if(project){
      if(project[field] == this._id){
        return true;
      }
    }
    return false;
```

```
  }
});

Template.selectManager.helpers({
  select: function(project, field){
    if(project){
      if(project[field] == this._id){
        return true;
      }
    }
    return false;
  }
});

Template.userInfo.helpers({
  itsMe: function(){
    if(this._id==Meteor.userId()){
      return true;
    }else{
      return false;
    }
  }
});

Template.userRow.helpers({
  itsMe: function(){
    if(this._id==Meteor.userId()){
      return true;
    }else{
      return false;
    }
  }
});
```

client/helpers/V2Helpers.js

```
Template.checkRow.helpers({
  equal: function(a, b){
    return (a==b);
  },
  color: function(){
    if(this.result=="OK"){
      return "#85e085"
    }else if(this.result=="NOK"){
      return "#ff6666"
    }else if(this.result=="POK"){
      return "#ffc266"
    }else if(!this.applicable){
      return "#c2c2a3"
    }
```

```
  }
});

Template.checklistInfo.helpers({
  findChecks: function(type){
    if(this.checks){
      checks=[];
      for(i=0;i<this.checks.length;i++){
        if(this.checks[i].type==type){
          checks.push(this.checks[i])
        }
      }
      return checks;
    }
  },
  svn:function(){
    toll=Tollgates.findOne({checklist:this._id});
    if(toll){
      return toll.link;
    }
  }
});

Template.checklistTable.helpers({
  findChecks: function(type){
    if(this.data){
      checks=[];
      for(i=0;i<this.data.checks.length;i++){
        if(this.data.checks[i].type==type){
          checks.push(this.data.checks[i])
        }
      }
      return checks;
    }
  }
});

Template.projectInfo.helpers({
  schemeName: function(Id){
    scheme = Schemes.findOne({_id:Id});
    if(scheme){
      return scheme.name;
    }
  }
});

Template.selectScheme.helpers({
  schemes: function(){
    return Schemes.find({});
  },
  select: function(project, field){
```

```
    if(project){
      if(project[field] == this._id){
        return true;
      }
    }
    return false;
  }
});

Template.schemesList.helpers({
  schemes: function(){
    return Schemes.find({});
  }
});

Template.newScheme.helpers({
  templates:function(Id){
    return Checklists.find({phase:Id, template:true})
  }
});

Template.schemeEdit.helpers({
  templates:function(Id){
    return Checklists.find({phase:Id, template:true})
  },
  equal:function(scheme, phase, phase2){
    if(scheme){
      return (scheme[phase]==phase2);
    }
  }
});

Template.checklistEdit.helpers({
  allChecks:function(type){
    //return Checks.find({type:type}, {sort:{num:1}});
    return Checks.find({type:type}, {sort:{_id:1}});
  },
  equal:function(phase, phase2){
    return (phase==phase2);
  },
  checked:function(Id, checks){
    if(checks){
      for(i=0;i<checks.length;i++){
        if(checks[i]._id==Id){
          return true
        }
      }
      return false
    }
  }
});
```

```
Template.checklistList.helpers({
  templates: function(phase){
    return Checklists.find({phase:phase, template:true});
  }
});
```

client/helpers/V3Helpers.js

```
Template.offers.helpers({
  offers: function(){
    val = Session.get("archivedOffers");
    return Offers.find({archived:val}, {sort:{customer:1}});
  },
  showArchived:function(){
    val = Session.get("archivedOffers");
    return val;
  }
});

Template.offerRow.helpers({
  color: function(){
    if(this.status=="Awarded"){
      return "#85e085"
    }else if(this.status=="Dropped"){
      return "#ff6666"
    }
  }
});

Template.KPIs.helpers({
  projects: function(){
    projects=Session.get("closedProjects");
    if (projects){
      return Projects.find({archived:true},{sort:{name:1}});
    }else{
      return Projects.find({archived:false},{sort:{name:1}});
    }
  },
  closed: function(){
    return Session.get("closedProjects");
  }
});

Template.lastKPI.helpers({
  kpi:function(){
    return KPIs.findOne({project:this._id}, {sort:{time:-1}});
  },
  getTargetData:function(parentData){
    return parentData[this._id];
```

```
  }
});

Template.KPIsTable.helpers({
  kpis:function(id){

    return KPIs.find({project:id}, {sort:{time:-1}});
  },
  getTargetData:function(parentData){
    return parentData[this._id];
  }
});

Template.editKPI.helpers({
  getTargetData:function(parentData){
    return parentData[this._id];
  }
});
```

client/helpers/V4Helpers.js

```
Template.newEvent.helpers({
  phases: function(){
    return Phases.find({project:this._id});
  },
  releases: function(){
    return ReleasesTime.find({project:this._id, phase:Session.get("phaseEvent")})
  }
});

Template.eventEdit.helpers({
  phases: function(){
    return Phases.find({project:this.project});
  },
  releases: function(){
    //console.log(document.getElementById('phase'));
    phase = Session.get("phaseEvent");
    return ReleasesTime.find({project:this.project, phase:phase})
  }
});

Template.newReleaseTime.helpers({
  phases: function(){
    return Phases.find({project:this._id});
  }
});

Template.releaseTimeInfo.helpers({
  phaseName: function(Id){
```

```
    phase = Phases.findOne(Id);
    if(phase){
      return phase.name;
    }
  }
});

Template.releaseTimeEdit.helpers({
  phases: function(){
    return Phases.find({project:this.project});
  }
});

Template.eventInfo.helpers({
  phaseName: function(Id){
    phase = Phases.findOne(Id);
    if(phase){
      return phase.name;
    }
  },
  categoryName: function(Id){
    category = Categories.findOne(Id);
    if(category){
      return category.name;
    }
  },
  releaseName: function(Id){
    release = ReleasesTime.findOne(Id);
    if(release){
      return release.name;
    }
  },
  typeName: function(Id){
    type = Types.findOne(Id);
    if(type){
      return type.name;
    }
  },
});
```

client/controllers/categoriesController.js

```
Template.categories.events({
  "click #new": function(event, template){
    swal({
      title: "New Category!",
      text: 'Category name:',
      type: 'input',
      showCancelButton: true,
      closeOnConfirm: false,
```

```
      }, function(name){
        if(name){
          target=Categories.findOne({name:name});
          if(target){
            swal.showInputError("Invalid name!");
          }else{
            Meteor.call("addCategory", name, function(error){
              if(error){
                console.log(error);
              }else{
                swal("Success!","Category created","success");
              }
            })
          }
        }else{
          swal.close();
        }
      }
    )
  }
});

Template.categoryInfo.events({
  "click #edit": function(event, template){
    Id=this._id;
    swal({
      title: "Edit Category!",
      text: 'Category name:',
      type: 'input',
      showCancelButton: true,
      closeOnConfirm: false,
      }, function(name){
        target=Categories.findOne({name:name});
        if(target){
          swal.showInputError("Invalid name!");
        }else{
          Meteor.call("updateCategory", Id,name, function(error){
            if(error){
              console.log(error);
            }else{
              swal("Success!","Category updated","success");
            }
          })
        }
      }
    )
  },
  "click #delete" : function(event, template){
    Id=this._id;

    swal({
```

```
      title: "Are you sure?",
      text: "",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      Meteor.call("deleteCategory", Id, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
        }else{
          swal(
            "Deleted!",
            "Your category has been deleted.",
            "success");
        }
      });
    });
  }
});
```

client/controllers/chartController.js

```
Template.chart.rendered = function(){
  var data=this.data;

  google.charts.setOnLoadCallback(drawChart(data));
};

function drawChart(id){
  kpis=KPIs.find({project:id}, {sort:{time:1}}).map(function(data){
    targets = Targets.find({}).map(function(data){return data._id});
    sample=[data.week];
    for(i=0;i<targets.length;i++){
      sample.push(data[targets[i]]);
    }
    return sample;
  });
  var targets = Targets.find({}).map(function(data){return data.name});

  if(google.visualization == undefined){
    document.location.reload(true);
  }else{
    var data = new google.visualization.DataTable();
    data.addColumn('number', 'Week');
    for(i=0;i<targets.length;i++){
      data.addColumn('number', targets[i]);
    }
```

```
    data.addRows(kpis);

    if(data.Lf.length!=0){
      var chart = new google.charts.Line(document.getElementById(id));
      chart.draw(data);
    }else{
      data = google.visualization.arrayToDataTable([
        ['Week', 'no data'],
        ['', 0],
      ]);

      options = {
        legend : {position: 'none'}
      };

      chart = new google.charts.Line(document.getElementById(id));
      chart.draw(data,options);
    }
  }
}
```

client/controllers/checklistController.js

```
Template.newChecklist.events({
  "click button": function(event, template){
    if(event.currentTarget.id=="cancel"){
      history.back();
    }else if (event.currentTarget.id=="save"){
      phase=template.find("#phase").value;
      name=template.find("#name").value;
      if(name==""){
        swal("Warning","Name is required!","warning");
      }else{
        var selected=[];
        $('input:checked').each(function(){
          selected.push($(this).attr('value'));
        })
        Meteor.call("createChecklist", name, phase, selected, function(error, result){
          if(error){
            swal("Error!",error,"error");
          }else{
            swal("Success!","Checklist created","success")
            Router.go('/checklist/templates')
          }
        });
      }
    }else{
      swal({
        title: "New Check!",
        text: 'Check ID:',
```

```
        type: 'input',
        inputType:"number",
        showCancelButton: true,
        closeOnConfirm: false,
        }, function(id){
          id=event.currentTarget.id+"."+id
          check=Checks.findOne({_id:id});
          if(check){
            swal.showInputError("Invalid ID!");
          }else{
            swal({
              title: "New Check!",
              text: 'Description:',
              type: 'input',
              showCancelButton: true,
              closeOnConfirm: false,
              closeOnCancel:false
              }, function(desc){
                if(desc){
                  Meteor.call("createCheck", id, desc,function(error, result){
                    if(error){
                      swal("Error!",error,"error")
                    }else{
                      swal("Success!","Check created","success")
                    }
                  });
                }else{
                  swal.close();
                }
              }
            )
          }
        }
      );
    }
  },
});

Template.checklistEdit.events({
  "click button": function(event, template){
    if(event.currentTarget.id=="cancel"){
      history.back();
    }else if (event.currentTarget.id=="save"){
      phase=template.find("#phase").value;
      name=template.find("#name").value;
      if(name==""){
        name=tempalte.data.name;
      }
      var selected=[];
      $('input:checked').each(function(){
        selected.push($(this).attr('value'));
```

```
      })
      Meteor.call("updateChecklist", template.data._id, name, phase, selected,
function(error, result){
        if(error){
          swal("Error!",error,"error");
        }else{
          swal("Success!","Checklist saved","success")
          Router.go('/checklist/templates')
        }
      });
    }else if (event.currentTarget.id=="delete"){
      var Id = template.data._id;
      swal({
        title: "Are you sure?",
        text: "You will not be able to recover this Checklist!",
        type: "warning",
        showCancelButton: true,
        confirmButtonColor: "#DD6B55",
        confirmButtonText: "Yes, delete it!",
        closeOnConfirm: false
        },
      function(){
        Meteor.call("deleteChecklist", Id, function(error, result){
          if(error){
            sweetAlert("Error", error.reason, "error");
          }else{
            swal(
              "Deleted!",
              "Your Checklist has been deleted.",
              "success");
              history.back();
          }
        });
      });
    }else{
      swal({
        title: "New Check!",
        text: 'Check ID:',
        type: 'input',
        inputType:"number",
        showCancelButton: true,
        closeOnConfirm: false,
        }, function(id){
          id=event.currentTarget.id+"."+id
          check=Checks.findOne({_id:id});
          if(check){
            swal.showInputError("Invalid ID!");
          }else{
            swal({
              title: "New Check!",
              text: 'Description:',
```

```
              type: 'input',
              showCancelButton: true,
              closeOnConfirm: false,
              }, function(desc){
                Meteor.call("createCheck", id, desc,function(error, result){
                  if(error){
                    swal("Error!",error,"error")
                  }else{
                    swal("Success!","Check created","success")
                  }
                });
              }
            )
          }
        }
      );
    }
  },
});

Template.checklistInfo.events({
  "click #cancel": function(event, template){
     history.back();
  },
  "click #save":function(event,template){
    this.assistants=template.find("#assistants").value;
    this.conclusions=template.find("#conclusions").value;
    Meteor.call("changeChecklist", this);
    swal("Success!","Checklist saved","success");
  }
});

Template.checklistTable.events({
  "click #save2": function(event, template){
    swal("Success!","Checklist saved","success");
  }
});
```

client/controllers/checksController.js

```
Template.newChecklist.events({
  "click button": function(event, template){
    if(event.currentTarget.id=="cancel"){
      history.back();
    }else if (event.currentTarget.id=="save"){
      phase=template.find("#phase").value;
      name=template.find("#name").value;
      if(name==""){
        swal("Warning","Name is required!","warning");
      }else{
```

```
        var selected=[];
        $('input:checked').each(function(){
          selected.push($(this).attr('value'));
        })
        Meteor.call("createChecklist", name, phase, selected, function(error, result){
          if(error){
            swal("Error!",error,"error");
          }else{
            swal("Success!","Checklist created","success")
            Router.go('/checklist/templates')
          }
        });
      }
    }else{
      swal({
        title: "New Check!",
        text: 'Check ID:',
        type: 'input',
        inputType:"number",
        showCancelButton: true,
        closeOnConfirm: false,
        }, function(id){
          id=event.currentTarget.id+"."+id
          check=Checks.findOne({_id:id});
          if(check){
            swal.showInputError("Invalid ID!");
          }else{
            swal({
              title: "New Check!",
              text: 'Description:',
              type: 'input',
              showCancelButton: true,
              closeOnConfirm: false,
              closeOnCancel:false
              }, function(desc){
                if(desc){
                  Meteor.call("createCheck", id, desc,function(error, result){
                    if(error){
                      swal("Error!",error,"error")
                    }else{
                      swal("Success!","Check created","success")
                    }
                  });
                }else{
                  swal.close();
                }
              }
            )
          }
        }
      );
```

```
    }
  },
});

Template.checklistEdit.events({
  "click button": function(event, template){
    if(event.currentTarget.id=="cancel"){
      history.back();
    }else if (event.currentTarget.id=="save"){
      phase=template.find("#phase").value;
      name=template.find("#name").value;
      if(name==""){
        name=tempalte.data.name;
      }
      var selected=[];
      $('input:checked').each(function(){
        selected.push($(this).attr('value'));
      })
      Meteor.call("updateChecklist", template.data._id, name, phase, selected,
function(error, result){
        if(error){
          swal("Error!",error,"error");
        }else{
          swal("Success!","Checklist saved","success")
          Router.go('/checklist/templates')
        }
      });
    }else if (event.currentTarget.id=="delete"){
      var Id = template.data._id;
      swal({
        title: "Are you sure?",
        text: "You will not be able to recover this Checklist!",
        type: "warning",
        showCancelButton: true,
        confirmButtonColor: "#DD6B55",
        confirmButtonText: "Yes, delete it!",
        closeOnConfirm: false
        },
      function(){
        Meteor.call("deleteChecklist", Id, function(error, result){
          if(error){
            sweetAlert("Error", error.reason, "error");
          }else{
            swal(
              "Deleted!",
              "Your Checklist has been deleted.",
              "success");
              history.back();
          }
        });
      });
```

```
    }else{
      swal({
        title: "New Check!",
        text: 'Check ID:',
        type: 'input',
        inputType:"number",
        showCancelButton: true,
        closeOnConfirm: false,
        }, function(id){
          id=event.currentTarget.id+"."+id
          check=Checks.findOne({_id:id});
          if(check){
            swal.showInputError("Invalid ID!");
          }else{
            swal({
              title: "New Check!",
              text: 'Description:',
              type: 'input',
              showCancelButton: true,
              closeOnConfirm: false,
              }, function(desc){
                Meteor.call("createCheck", id, desc,function(error, result){
                  if(error){
                    swal("Error!",error,"error")
                  }else{
                    swal("Success!","Check created","success")
                  }
                });
              }
            )
          }
        }
      );
    }
  },
});

Template.checklistInfo.events({
  "click #cancel": function(event, template){
     history.back();
  },
  "click #save":function(event,template){
    this.assistants=template.find("#assistants").value;
    this.conclusions=template.find("#conclusions").value;
    Meteor.call("changeChecklist", this);
    swal("Success!","Checklist saved","success");
  }
});

Template.checklistTable.events({
  "click #save2": function(event, template){
```

```
    swal("Success!","Checklist saved","success");
  }
});
```

client/controllers/eventsController.js

```
Template.newEvent.rendered = function(){
  $('#date').datepicker({
    weekStart: 1,
    format: "yyyy-mm-dd",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}

Template.eventEdit.rendered = function(){
  $('#date').datepicker({
    weekStart: 1,
    format: "yyyy-mm-dd",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}

Template.newEvent.events({
  "click #cancel":function(event,template){
    history.back();
  },
  "click #save": function(event, template){
    name=template.find('#name').value;
    date=template.find('#date').value;
    phase=template.find('#phase').value;
    result=template.find('#result').value;
    category=template.find('#category').value;
    rel=template.find('#release').value;
    type=template.find('#type').value;


    if(name=="" || date==""){
      sweetAlert("Warning", "Name and Date are required!", "warning");
    }else{
      Meteor.call("addEvent", name, date, rel, phase, result, category, this._id, type,
function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
```

```
        }else{
          history.back();
        }
      });
    }
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  },
  "change #phase":function(event, template){
    Session.set("phaseEvent", event.target.value);
  }
});

Template.eventEdit.events({
  "click #cancel":function(event,template){
    history.back();
  },
  "click #save": function(event, template){
    name=template.find('#name').value;
    date=template.find('#date').value;
    phase=template.find('#phase').value;
    result=template.find('#result').value;
    type=template.find('#type').value;
    category=template.find('#category').value;
    release=template.find('#release').value;

    if(name==""){
        name=this.name;
    }
    if(date==""){
      sweetAlert("Warning", "Date is required!", "warning");
    }else{
      Meteor.call("updateEvent",this._id, name, date, phase, release,result, category,
type, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
        }else{
          history.back();
        }
      });
    }
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  },
```

```
  "change #phase":function(event, template){
    Session.set("phaseEvent", event.target.value);
  },
  "click #delete":function(event, template){
    Meteor.call("deleteEvent", this._id, function(error,result){
      if(error){
        console.log(error);
      }else{
        history.back();
        history.back();
      }
    })
  }
});
```

client/controllers/homeController.js

```
Template.home.events({
  "click #switch-to-run": function(event, template){
    Session.set("closedProjects", false);
  },
  "click #switch-to-close": function(event,template){
    Session.set("closedProjects", true);
  }
});
```

client/controllers/kpiController.js

```
Template.newKPI.rendered = function(){
  $('#date').datepicker({
    weekStart: 1,
    calendarWeeks: true,
    daysOfWeekHighlighted: "1",
    daysOfWeekDisabled: "0,2,3,4,5,6",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}

Template.editKPI.rendered = function(){
  $('#date').datepicker({
    weekStart: 1,
    calendarWeeks: true,
    daysOfWeekHighlighted: "1",
    daysOfWeekDisabled: "0,2,3,4,5,6",
    maxViewMode: 0,
    todayBtn: "linked",
```

```
    autoclose: true,
    todayHighlight: true
  });
}

Template.newKPI.events({
  "click #save": function(event, template){
    date=template.find('#date').value;
    if (date==""){
      console.log("Error: date undefined");
    }else{
      time= new Date(date);
      time2 = time;
      time.setHours(0,0,0);
      time.setDate(time.getDate() + 4 - (time.getDay()||7));
      var yearStart = new Date(time.getFullYear(),0,1);
      var weekNo = Math.ceil(( ( (time - yearStart) / 86400000) + 1)/7);

      kpi={date:date, week:weekNo, year:time.getFullYear() , time:time2.getTime(),
project:template.data._id};

      Targets.find({}).forEach(function(target){
        value=template.find('#'+target._id).value;
        if(value==""){
          value=0;
        }else{
          value=parseInt(value);
        }
        kpi[target._id]=value
      })

      Meteor.call("addKPI", kpi, function(error){
        if(error){
          sweetAlert("Error", error.reason, "error");
        }else{
          history.back();
        }
      });
    }
  },
  "click #cancel":function(event,template){
    history.back();
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  }
});
```

```
Template.editKPI.events({
  "click #save": function(event, template){
    date=template.find('#date').value;
    if (date==""){
      console.log("Error: date undefined");
    }else{
      time= new Date(date);
      time2 = time;
      time.setHours(0,0,0);
      time.setDate(time.getDate() + 4 - (time.getDay()||7));
      var yearStart = new Date(time.getFullYear(),0,1);
      var weekNo = Math.ceil(( ( (time - yearStart) / 86400000) + 1)/7);

      kpi={date:date, week:weekNo, year:time.getFullYear() , time:time2.getTime(),
project:template.data.project};

      Targets.find({}).forEach(function(target){
        value=template.find('#'+target._id).value;
        if(value==""){
          value=0;
        }else{
          value=parseInt(value);
        }
        kpi[target._id]=value
      })
      Meteor.call("updateKPI", template.data._id, kpi, function(error){
        if(error){
          sweetAlert("Error", error.reason, "error");
        }else{
          history.back();
        }
      });
    }
  },
  "click #cancel":function(event,template){
    history.back();
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  },
  "click #delete":function(event,template){
    Id=this._id;

    swal({
      title: "Are you sure?",
      text: "",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
```

```
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      swal(
        "Deleted!",
        "Your KPI has been deleted.",
        "success");
      history.back();
      Meteor.call("deleteKPI", Id, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
        }
      });
    });
  }
});

Template.KPIs.events({
  "click #switch-to-run": function(event, template){
    Session.set("closedProjects", false);
  },
  "click #switch-to-close": function(event,template){
    Session.set("closedProjects", true);
  }
});


Template.targets.events({
  "click #new": function(event, template){
    swal({
      title: "New Target!",
      text: 'Target name:',
      type: 'input',
      showCancelButton: true,
      closeOnConfirm: false,
      }, function(name){
        if(name){
          target=Targets.findOne({name:name});
          if(target){
            swal.showInputError("Invalid name!");
          }else{
            Meteor.call("addTarget", name, function(error){
              if(error){
                console.log(error);
              }else{
                swal("Success!","Target created","success");
              }
            })
          }
        }else{
```

```
          swal.close();
        }
      }
    )
  }
});

Template.targetInfo.events({
  "click #edit": function(event, template){
    Id=this._id;
    swal({
      title: "Edit Target!",
      text: 'Target name:',
      type: 'input',
      showCancelButton: true,
      closeOnConfirm: false,
      }, function(name){
        target=Targets.findOne({name:name});
        if(target){
          swal.showInputError("Invalid name!");
        }else{
          Meteor.call("updateTarget", Id,name, function(error){
            if(error){
              console.log(error);
            }else{
              swal("Success!","Target updated","success");
            }
          })
        }
      }
    )
  },
  "click #delete" : function(event, template){
    Id=this._id;

    swal({
      title: "Are you sure?",
      text: "",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      Meteor.call("deleteTarget", Id, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
        }else{
          swal(
            "Deleted!",
```

```
            "Your target has been deleted.",
            "success");
        }
      });
    });
  }
});
```

client/controllers/offersController.js

```
Template.offers.events({
  "click #archived": function(event, template){
     Session.set("archivedOffers", true);
  },
  "click #unarchived":function(event, template){
    Session.set("archivedOffers", false);
  }
});

Template.newOffer.events({
  "click #save": function(event, template){
    customer=template.find("#cust").value;
    if(customer==""){
      console.log("error");
    }else{
      bu=template.find("#bu").value;
      product=template.find("#prod").value;
      scope=template.find("#scope").value;
      status=template.find("#status").value;
      svn=template.find("#svn").value;
      comments=template.find("#comment").value;
      Meteor.call("addOffer", customer, bu, product, scope, status, svn, comments,
function(error){
        if(error){
          console.log(error);
        }else{
          history.back();
        }
      });
    }
  },
  "focus #svn":function(event, template){
    template.find("#svn").select();
  }
});

Template.offerInfo.events({
  "click #save": function(event, template){
    id=template.data._id;
    customer=template.find("#cust").value;
```

```
    bu=template.find("#bu").value;
    product=template.find("#prod").value;
    scope=template.find("#scope").value;
    status=template.find("#status").value;
    svn=template.find("#svn").value;
    comments=template.find("#comment").value;
    Meteor.call("updateOffer", id, customer, bu, product, scope, status, svn, comments,
function(error){
      if(error){
        console.log(error);
      }else{
        history.back();
      }
    });
  },
  "focus #svn":function(event, template){
    template.find("#svn").select();
  },
  "click #delete":function(event, template){
    Meteor.call("deleteOffer", template.data._id, function(error){
      if(error){
        console.log(error);
      }else{
        history.back();
      }
    })
  }
});

Template.offerRow.events({
  "change #status": function(event, template){
    Meteor.call("updateStatusOffer", template.data._id, event.target.value,
function(error){
      if(error){
        console.log(error);
      }
    })
  },
  "focusout #comment":function(event, template){
    Meteor.call("updateCommentOffer", template.data._id, event.target.value,
function(error){
      if(error){
        console.log(error);
      }
    })
  },
  "click #switch":function(event,template){
    Meteor.call("switchArchiveOffer", template.data._id)
  }
});
```

client/controllers/phasesController.js

```
Template.newPhase.rendered = function(){
  $('#datepicker').datepicker({
    weekStart: 1,
    format: "yyyy-mm-dd",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}
Template.phaseEdit.rendered = function(){
  $('#datepicker').datepicker({
    weekStart: 1,
    format: "yyyy-mm-dd",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}

Template.newPhase.events({
  "click #cancel":function(event,template){
    history.back();
  },
  "click #save": function(event, template){
    name=template.find('#name').value;
    start=template.find('#start').value;
    end=template.find('#end').value;

    if(name=="" || start=="" || end==""){
      sweetAlert("Warning", "Name and Date Range are required!", "warning");
    }else{
      Meteor.call("addPhase", this._id, name, start, end, true, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
        }else{
          history.back();
        }
      });
    }
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  }
});
```

```
Template.phaseEdit.events({
  "click #cancel":function(event,template){
    history.back();
  },
  "click #save": function(event, template){
    name=template.find('#name').value;
    start=template.find('#start').value;
    end=template.find('#end').value;

    if(name==""){
      name=this.name;
    }
    if(start=="" || end==""){
      sweetAlert("Warning", "Date Range is required!", "warning");
    }else{
      Meteor.call("updatePhase",this._id, name, start, end, true, function(error,
result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
        }else{
          history.back();
        }
      });
    }
  },
  "click #switch":function(event,tempalte){
    Meteor.call("updatePhase",this._id, this.name, this.start, this.end, !this.open,
function(error, result){
      if(error){
        sweetAlert("Error", error.reason, "error");
        console.log(error);
      }else{
        history.back();
      }
    });
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  },
  "click #delete":function(event, template){
    Meteor.call("deletePhase", this._id, function(error,result){
      if(error){
        console.log(error);
      }else{
        history.back();
        history.back();
```

```
      }
    })
  }
});
```

client/controllers/projectsController.js

```
Template.projectPage.events({
  "click #add-release": function(event, template){
    Meteor.call("addRelease", template.data._id);
  }
});

Template.createProject.events({
  "click #add": function(event, template){
    name=template.find('#name').value;
    ref=template.find('#ref').value.replace(/ /g,"_");
    projectLead=template.find('#projectLead').value;
    sw=template.find('#swLead').value;
    hw=template.find('#hwLead').value;
    mech=template.find('#mechLead').value;
    system=template.find('#system').value;
    qa=template.find('#qa').value;
    tv=template.find('#tv').value;
    tr=template.find('#tr').value;
    compliance=template.find('#compliance').value;
    scheme=template.find('#scheme').value;

    if(name=="" || ref==""){
      sweetAlert("Warning", "Name and Reference are required!", "warning");
    }else{
      Meteor.call("createProject", name, ref, projectLead, sw, hw, mech, system, qa, tv,
tr, compliance, scheme, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
        }else{
          Router.go('/project/'+ref);
        }
      });
    }
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#add').click();
    }
  }
});

Template.projectEdit.events({
```

```
  "click #save": function(event, template){
    id =  template.data._id;
    projectLead=template.find('#projectLead').value;
    sw=template.find('#swLead').value;
    hw=template.find('#hwLead').value;
    mech=template.find('#mechLead').value;
    system=template.find('#system').value;
    scheme=template.find('#scheme').value;
    qa=template.find('#qa').value;
    tv=template.find('#tv').value;
    tr=template.find('#tr').value;
    compliance=template.find('#compliance').value;

    Meteor.call("updateProject", id, projectLead, sw, hw, mech, system, qa, tv, tr,
compliance, scheme, function(error, result){
      if(error){
        sweetAlert("Error", error.reason, "error");
        console.log(error);
      }else{
        Router.go('/project/'+template.data._id);
      }
    });
  },
  "click #delete":function(event,template){
    swal({
      title: "Are you sure?",
      text: "You will not be able to recover this project!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      Meteor.call("deleteProject", template.data._id, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
        }else{
          swal(
            "Deleted!",
            "Your project has been deleted.",
            "success");
          Router.go('/');
        }
      });
    });
  },
  "click #close":function(event, template){
    Meteor.call("closeProject",template.data._id, function(error, result){
      if(error){
```

```
        console.log("error", error);
      }
      else{
        swal("Closed!","","success");
        history.back();
      }
    });
  },
  "click #open":function(event, template){
    Meteor.call("openProject",template.data._id, function(error, result){
      if(error){
        console.log("error", error);
      }
      else{
        swal("Opened!","","success");
        history.back();
      }
    });
  },
  "click #cancel":function(event, template){
    history.back();
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  }
});

Template.projectTable.events({
  "click #hide-all": function(event, template){
      Session.set("hide", Projects.find().map(function(project){return project._id}));
      Session.set("showAll", 'true');
  },
  "click #show-all": function(event, template){
      Session.set("hide", []);
      Session.set("showAll", 'false');
  },
  "click #hide-toll": function(event, template){
      Session.set("tollgates", 'true');
  },
  "click #show-toll": function(event, template){
      Session.set("tollgates", 'false');
  }
});

Template.projectRow.events({
  "click #hide": function(event, template){
     list = Session.get("hide");
     list.push(template.data._id);
     Session.set("hide", list);
```

```
  },
  "click #show": function(event, template){
     list = Session.get("hide");
     list.splice(list.indexOf(template.data._id),1);
     Session.set("hide", list);
  }
});
```

client/controllers/releaseController.js

```
Template.releaseRow.events({
  "click #delete": function(event, template){
    relId = this._id;
    swal({
      title: "Are you sure?",
      text: "You will not be able to recover this release!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      Meteor.call("deleteRelease", relId, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
        }else{
          swal(
            "Deleted!",
            "Your release has been deleted.",
            "success");
        }
      });
    });
  }
});
```

client/controllers/releaseTimeController.js

```
Template.newReleaseTime.rendered = function(){
  $('#datepicker').datepicker({
    weekStart: 1,
    format: "yyyy-mm-dd",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}
```

```
Template.releaseTimeEdit.rendered = function(){
  $('#datepicker').datepicker({
    weekStart: 1,
    format: "yyyy-mm-dd",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}

Template.newReleaseTime.events({
  "click #cancel":function(event,template){
    history.back();
  },
  "click #save": function(event, template){
    name=template.find('#name').value;
    phase=template.find('#phase').value;
    start=template.find('#start').value;
    end=template.find('#end').value;

    if(name=="" || start=="" || end==""){
      sweetAlert("Warning", "Name and Date Range are required!", "warning");
    }else{
      Meteor.call("addReleaseTime", phase, name, start, end, this._id, function(error,
result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
        }else{
          history.back();
        }
      });
    }
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  }
});


Template.releaseTimeEdit.events({
  "click #cancel":function(event,template){
    history.back();
  },
  "click #save": function(event, template){
    name=template.find('#name').value;
```

```
    phase=template.find('#phase').value;
    start=template.find('#start').value;
    end=template.find('#end').value;

    if(name==""){
      name=this.name;
    }
    if(start=="" || end==""){
      sweetAlert("Warning", "Date Range is required!", "warning");
    }else{
      Meteor.call("updateReleaseTime",this._id, phase, name, start, end, function(error,
result){
        if(error){
          sweetAlert("Error", error.reason, "error");
          console.log(error);
        }else{
          history.back();
        }
      });
    }
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  },
  "click #delete":function(event, template){
    Meteor.call("deleteReleaseTime", this._id, function(error,result){
      if(error){
        console.log(error);
      }else{
        history.back();
        history.back();
      }
    })
  }
});
```

client/controllers/schemesController.js

```
Template.newScheme.events({
  "click button": function(event, template){
    if(event.currentTarget.id=="cancel"){
      history.back();
    }else if (event.currentTarget.id=="save"){
      name=template.find("#name").value;
      phase3FL=template.find("#phase3FL").value;
      phase3CL=template.find("#phase3CL").value;
      phase4FL=template.find("#phase4FL").value;
      phase4CL=template.find("#phase4CL").value;
```

```
      phase5FL=template.find("#phase5FL").value;
      phase5CL=template.find("#phase5CL").value;
      phase6FL=template.find("#phase6FL").value;
      phase6CL=template.find("#phase6CL").value;

      Meteor.call("createScheme", name, phase3FL, phase3CL, phase4FL, phase4CL, phase5FL,
phase5CL, phase6FL, phase6CL, function(error, result){
        if(error){
          swal("Error!",error,"error");
        }else{
          swal("Success!","Scheme created","success")
          Router.go('/checklist/schemes')
        }
      });
    }
  },
});

Template.schemeEdit.events({
  "click #cancel": function(event, template){
     history.back();
  },
  "click #delete":function(event,template){
    Id = this._id;
    swal({
      title: "Are you sure?",
      text: "You will not be able to recover this scheme!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      Meteor.call("deleteScheme", Id, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
        }else{
          swal(
            "Deleted!",
            "Your scheme has been deleted.",
            "success");
            history.back();
        }
      });
    });
  },
  "click #save": function(event,template){
    name=template.find("#name").value;
    if (name==""){
      name=template.data.name;
```

```
    }
    phase3FL=template.find("#phase3FL").value;
    phase3CL=template.find("#phase3CL").value;
    phase4FL=template.find("#phase4FL").value;
    phase4CL=template.find("#phase4CL").value;
    phase5FL=template.find("#phase5FL").value;
    phase5CL=template.find("#phase5CL").value;
    phase6FL=template.find("#phase6FL").value;

    Meteor.call("updateScheme", template.data._id, name, phase3FL, phase3CL, phase4FL,
phase4CL, phase5FL, phase5CL, phase6FL, function(error, result){
      if(error){
        swal("Error!",error,"error");
      }else{
        swal("Success!","Scheme saved","success")
        Router.go('/checklist/schemes')
      }
    });
  }
});
```

client/controllers/timelineController.js

```
Template.timeline.onCreated(function(){
  this.subscribe("events");
  this.subscribe("releasesTime");
  this.subscribe("phases");
});

Template.timelineScheme.rendered = function(){
  var container = document.getElementById('visualization');

  var groups = new vis.DataSet();
  groups.add({id: 0, content: "Phases"});
  groups.add({id: 1, content: "Releases"});
  groups.add({id: 2, content: "Events"});

  var projectId = document.URL.split('/')[4];

  phases=Phases.find({project:projectId}).map(function(data){
    return {id: data._id, content: data.name, start: data.start, end: data.end, group:0}
  });

  rel = ReleasesTime.find({project:projectId}).map(function(data){
    return {id:data._id, content:data.name,start:data.start, end:data.end, group:1}
  });

  events=Events.find({project:projectId}).map(function(data){
    return {id: data._id, content: data.name, start: data.date, group:2}
```

```
  });

  data=phases.concat(rel, events);

  var items = new vis.DataSet(data)

 var options = {
 };

 // Create a Timeline
 var timeline = new vis.Timeline(container);
 timeline.setOptions(options);
 timeline.setGroups(groups);
 timeline.setItems(items);

 timeline.on('select', function (object) {
   if(object.items.length>0){
     Router.go("/project"+document.URL.split("project")[1]+"/"+object.items[0])
   }
 });
};
```

client/controllers/tollgatesController.js

```
Template.tollgateInfo.rendered = function(){
  $('#date').datepicker({
    weekStart: 1,
    format: "dd/mm/yyyy",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}

Template.newTollgate.rendered = function(){
  $('#date').datepicker({
    weekStart: 1,
    format: "dd/mm/yyyy",
    maxViewMode: 0,
    todayBtn: "linked",
    autoclose: true,
    todayHighlight: true
  });
}

Template.newTollgate.events({
  "click #save": function(event, template){
    date=template.find('#date').value;
```

```
    link=template.find('#linkSVN').value;
    result=template.find('#result').value;
    if (date=="" && result!="Undefined"){
      result="Undefined";
    }
    Meteor.call("addTollgate", this.release, this.phase, date, link, result,
function(error, result){
      if(error){
        sweetAlert("Error", error.reason, "error");
      }else{
        history.back();
      }
    });
  },
  "click #cancel":function(event,template){
    history.back();
  },
  "focus #linkSVN":function(event, template){
    template.find("#linkSVN").select();
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  }
});

Template.tollgateInfo.events({
  "click #save": function(event, template){
    date=template.find('#date').value;
    link=template.find('#linkSVN').value;
    result=template.find('#result').value;

    Meteor.call("updateTollgate", this._id, date, link, result, function(error, result){
      if(error){
        console.log("error", error);
      }else{
        history.back();
      }
    });
  },
  "click #delete":function(event,template){
    tollId=this._id
    swal({
      title: "Are you sure?",
      text: "You will not be able to recover this milestone!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
```

```
      },
    function(){
        Meteor.call("deleteTollgate", tollId, function(error, result){
        if(error){
          console.log("error", error);
        }else{
          swal(
            "Deleted!",
            "Your milestone has been deleted.",
            "success");
            history.back();
        }
      });
    });
  },
  "click #cancel":function(event,template){
    history.back();
  },
  "focus #link":function(event, template){
    template.find("#linkSVN").select();
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  }
});
```

client/controllers/typesController.js

```
Template.types.events({
  "click #new": function(event, template){
    swal({
      title: "New Type!",
      text: 'Type name:',
      type: 'input',
      showCancelButton: true,
      closeOnConfirm: false,
      }, function(name){
        if(name){
          target=Categories.findOne({name:name});
          if(target){
            swal.showInputError("Invalid name!");
          }else{
            Meteor.call("addType", name, function(error){
              if(error){
                console.log(error);
              }else{
                swal("Success!","Type created","success");
              }
```

```
            })
          }
        }else{
          swal.close();
        }
      }
    )
  }
});

Template.typeInfo.events({
  "click #edit": function(event, template){
    Id=this._id;
    swal({
      title: "Edit Type!",
      text: 'Type name:',
      type: 'input',
      showCancelButton: true,
      closeOnConfirm: false,
      }, function(name){
        target=Categories.findOne({name:name});
        if(target){
          swal.showInputError("Invalid name!");
        }else{
          Meteor.call("updateType", Id,name, function(error){
            if(error){
              console.log(error);
            }else{
              swal("Success!","Type updated","success");
            }
          })
        }
      }
    )
  },
  "click #delete" : function(event, template){
    Id=this._id;

    swal({
      title: "Are you sure?",
      text: "",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      Meteor.call("deleteType", Id, function(error, result){
        if(error){
          sweetAlert("Error", error.reason, "error");
```

```
        }else{
          swal(
            "Deleted!",
            "Your type has been deleted.",
            "success");
        }
      });
    });
  }
});
```

client/controllers/usersController.js

```
Template.usersList.onCreated(function(){
  this.subscribe("users");
});

Template.usersLogIn.events({
  "click #cancel": function(event, template){
     Router.go('/');
  },
  "click #login":function(event, template){
    username = template.find('#username').value;
    password = template.find('#password').value;
    Meteor.loginWithPassword(username, password, function(error){
      if(error){
        sweetAlert("Error", error.reason, "error");
      }else{
        history.back();
      }
    })
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#login').click();
    }
  }
});
Template.layout.events({
  "click #logout": function(event, template){
     Meteor.logout();
  }
});

Template.navbar.events({
  "click #logout": function(event, template){
     Meteor.logout();
  },
  "click #link":function(event,template){
    Router.go("/users/"+Meteor.userId());
```

```
  },
  "click #login":function(event, template){
    Router.go("/login");
  }
});


Template.createUser.events({
  "click #add": function(event, template){
    manager=template.find('#manager').checked;
    pass1 =template.find('#password').value;
    pass2 =template.find('#password2').value;
    admin =template.find('#admin').checked;
    if (pass1 != pass2){
      sweetAlert("Warning", "Passwords are not equal!", "warning");
    }else{
      username=template.find('#username').value;
      fullName=template.find('#full-name').value;
      if (pass1=="" || username =="" || fullName==""){
        sweetAlert("Warning", "Some fields are still empty!", "warning");
      }else{
        var userObject = {
          username: username,
          password: pass1,
          profile:{
            admin:admin,
            name:fullName,
            manager:manager
          }
        };
        Meteor.call("newUser", userObject, function(error, result){
          if(error){
            sweetAlert("Warning", error.reason, "warning");
          }else{
            Router.go('/users');
          }
        });
      }
    }
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#add').click();
    }
  }
});

Template.userRow.events({
  "click #delete": function(event, template){
     id = template.data._id;
     swal({
```

```
      title: "Are you sure?",
      text: "You will not be able to recover this user!",
      type: "warning",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, delete it!",
      closeOnConfirm: false
      },
    function(){
      Meteor.users.remove({_id:id});
      swal(
        "Deleted!",
        "Your user has been deleted.",
        "success");
    });
  }
});

Template.userInfo.events({
  "click #changePsw": function(event, template){
    id = template.data._id;
    swal({
      title: "Change Password",
      inputType: "password",
      inputPlaceholder: "Password",
      type: "input",
      showCancelButton: true,
      confirmButtonColor: "#DD6B55",
      confirmButtonText: "Yes, change it!",
      closeOnConfirm: false
      },
    function(inputValue){
      if(inputValue==""){
        swal.showInputError("You need to write something!");
      }else{
        Meteor.call("changePsw", id, inputValue, function(error, result){
          if(error){
            sweetAlert("Warning", error.reason, "warning");
          }else{
            swal(
              "Changed!",
              "The password has been changed.",
              "success");
          }
        });
      }
    });
  },
  "click #save":function(event, template){
    username=template.find('#username').value;
    admin=template.find('#admin').checked;
```

```
    name=template.find('#full-name').value;
    manager=template.find('#manager').checked;
    Meteor.users.update({_id:this._id}, {$set:{
      username:username,
      "profile.name":name,
      "profile.manager":manager,
      "profile.admin":admin
    }});
    swal("Success!","The user has been saved.","success");
    history.back();
  },
  "click #cancel":function(event, template){
    history.back();
  },
  "keypress":function(event, template){
    if(event.which==13){
      $('#save').click();
    }
  }
});
```

client/controllers/yearController.js

```
Template.drop.events({
  "click a": function(event, template){
     Session.set("year", event.currentTarget.innerText);
  }
});
```